IN THE UNITED STATES DISTRICT COURT
FOR THE DISTRICT OF DELAWARE

CIF LICENSING, LLC, d/b/a GE LICENSING,	)	C.A. No. 07-170 (JJF)
Plaintiff,	)	
v.	)	
AGERE SYSTEMS INC.,	)	
Defendant.	)	

APPENDIX OF EXHIBITS TO PLAINTIFF CIF LICENSING, LLC, d/b/a GE LICENSING'S OPENING CLAIM CONSTRUCTION BRIEF

VOLUME 1 – EXHIBITS A-E

OF COUNSEL:

Joel M. Freed
Brian E. Ferguson
Michael W. Connelly
McDermott Will & Emery LLP
600 13th Street, N.W.
Washington, DC 20005-3096
(202) 756-8327

Edwin H. Wheeler
McDermott Will & Emery LLP
3150 Porter Drive
Palo Alto, CA 94304-1212
(650) 813-5000

Richard L. Horwitz (#2246)
Philip A. Rovner (#3215)
David E. Moore (#3983)
POTTER ANDERSON & CORROON LLP
Hercules Plaza
P. O. Box 951
Wilmington, Delaware 19899
(302) 984-6000
rhorwitz@potteranderson.com
provner@potteranderson.com
dmoore@potteranderson.com

Attorneys for Plaintiff
CIF Licensing, LLC, d/b/a
GE Licensing

Dated: April 28, 2008

IN THE UNITED STATES DISTRICT COURT
FOR THE DISTRICT OF DELAWARE

CERTIFICATE OF SERVICE

I, Philip A. Rovner, hereby certify that on April 28, 2008, the within document was filed with the Clerk of the Court using CM/ECF; that the document was served on the following party as indicated; and that the document is available for viewing and downloading from CM/ECF.

BY HAND DELIVERY AND E-MAIL

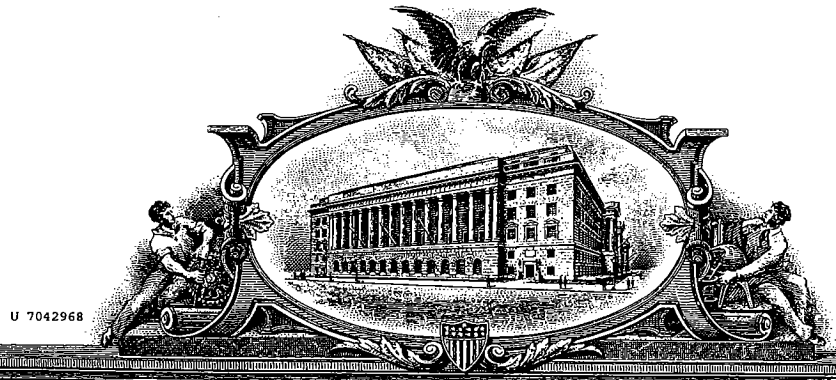
Josy W. Ingersoll, Esq.
John W. Shaw, Esq.
Young Conaway Stargatt & Taylor, LLP
The Brandywine Building
1000 West Street, 17th Floor
Wilmington, DE 19801

I hereby certify that on April 28, 2008 I have sent by E-mail the foregoing document to the following non-registered participants:

David E. Sipiora, Esq.
Ian L. Saffer, Esq.
Chad E. King, Esq.
Ryan D. Phillips, Esq.
Townsend and Townsend and Crew LLP
1200 17th Street, Suite 2700
Denver, CO 80202
desipiora@townsend.com
ilsaffer@townsend.com
ceking@townsend.com
rdphillips@townsend.com

/s/ Philip A. Rovner
Philip A. Rovner (#3215)
Potter Anderson & Corroon LLP
Hercules Plaza
P. O. Box 951
Wilmington, DE 19899
(302) 984-6000
provner@potteranderson.com

EXHIBIT A



U 7042968

THE UNITED STATES OF AMERICA

TO ALL TO WHOM THESE PRESENTS SHALL COME:

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office

December 08, 2006

THIS IS TO CERTIFY THAT ANNEXED HERETO IS A TRUE COPY FROM THE RECORDS OF THIS OFFICE OF:

U.S. PATENT: *5,048,054*
ISSUE DATE: *September 10, 1991*

By Authority of the
Under Secretary of Commerce for Intellectual Property
and Director of the United States Patent and Trademark Office

E. BORNETT
Certifying Officer

GE 000001

United States Patent [19]

Eyuboglu et al.

[11] **Patent Number: 5,048,054**

[45] **Date of Patent: Sep. 10, 1991**

[54] **LINE PROBING MODEM**

[75] Inventors: **Vedat M. Eyuboglu**, Boston; **Ping Dong**, Norwood, both of Mass.

[73] Assignee: **Codex Corporation**, Mansfield, Mass.

[21] Appl. No.: **351,199**

[22] Filed: **May 12, 1989**

[51] **Int. Cl.**[5] **H04B 7/10**

[52] **U.S. Cl.** **375/8**; 375/10; 375/38; 375/100; 455/135; 455/62

[58] **Field of Search** 375/58, 8, 10, 38, 100; 455/62, 135–137; 370/79; 340/825.03

[56] **References Cited**

U.S. PATENT DOCUMENTS

4,309,773	1/1982	Johnson et al.	455/62
4,630,126	12/1986	Kaku et al.	358/280
4,679,227	7/1987	Hughes-Hartogs	379/98
4,756,007	7/1988	Qureshi et al.	375/37

OTHER PUBLICATIONS

Eyuboglu, "Detection of Severely Distorted Signals Using Decision Feedback Noise Prediction with Interleaving," IEEE Transaction of Communications, vol. 36, No. 4, Apr. 1988, pp. 401–409.

USSN 351,186, "Trellis Precoding for Modulation Systems".

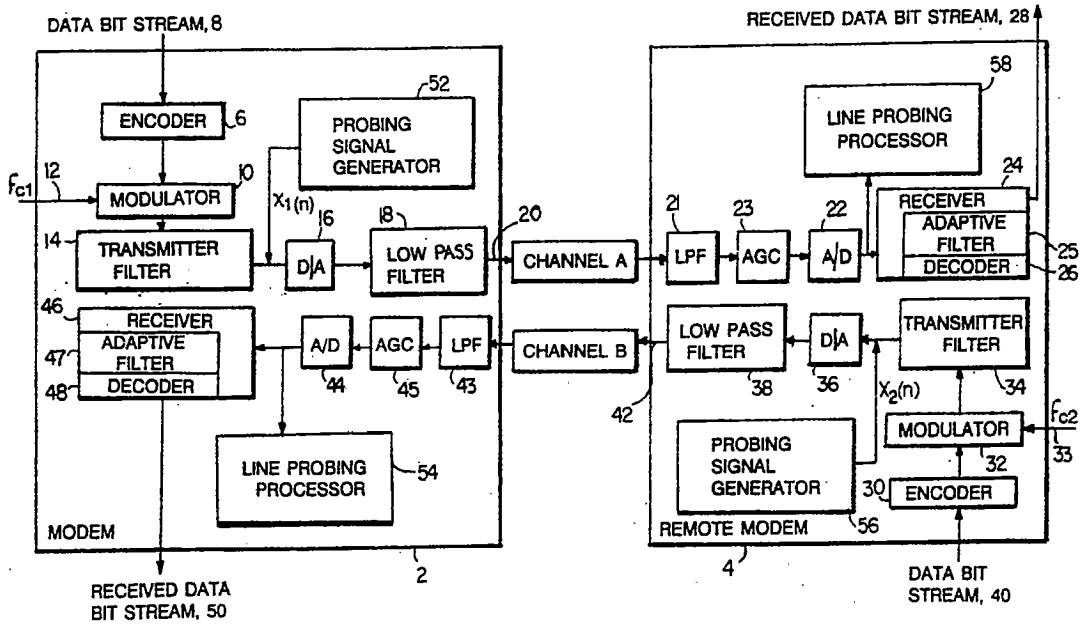
Kalet, "The Multitone Channel," IEEE Transactions on Communications, vol. 37, No. 2, Feb. 1989, pp. 119–124.

Primary Examiner—Benedict V. Safourek
Assistant Examiner—Tesfaldet Bocure
Attorney, Agent, or Firm—Fish & Richardson

[57] **ABSTRACT**

A modem for receiving data sent from a remote device over a communication channel by using a single carrier modulated signal, the modem including a receiver for receiving the modulated signal and for receiving a line probing signal sent by the remote device over the channel, the receiver being capable of receiving the modulated signal over any one of a plurality of frequency bands; a line probing processor for measuring characteristics of the channel based upon the received line probing signal; and a selector for selecting one of the plurality of frequency bands, said selection being based upon the measured characteristics of the channel, said selected frequency band to be used for receiving the modulated signal from the remote device.

71 Claims, 3 Drawing Sheets



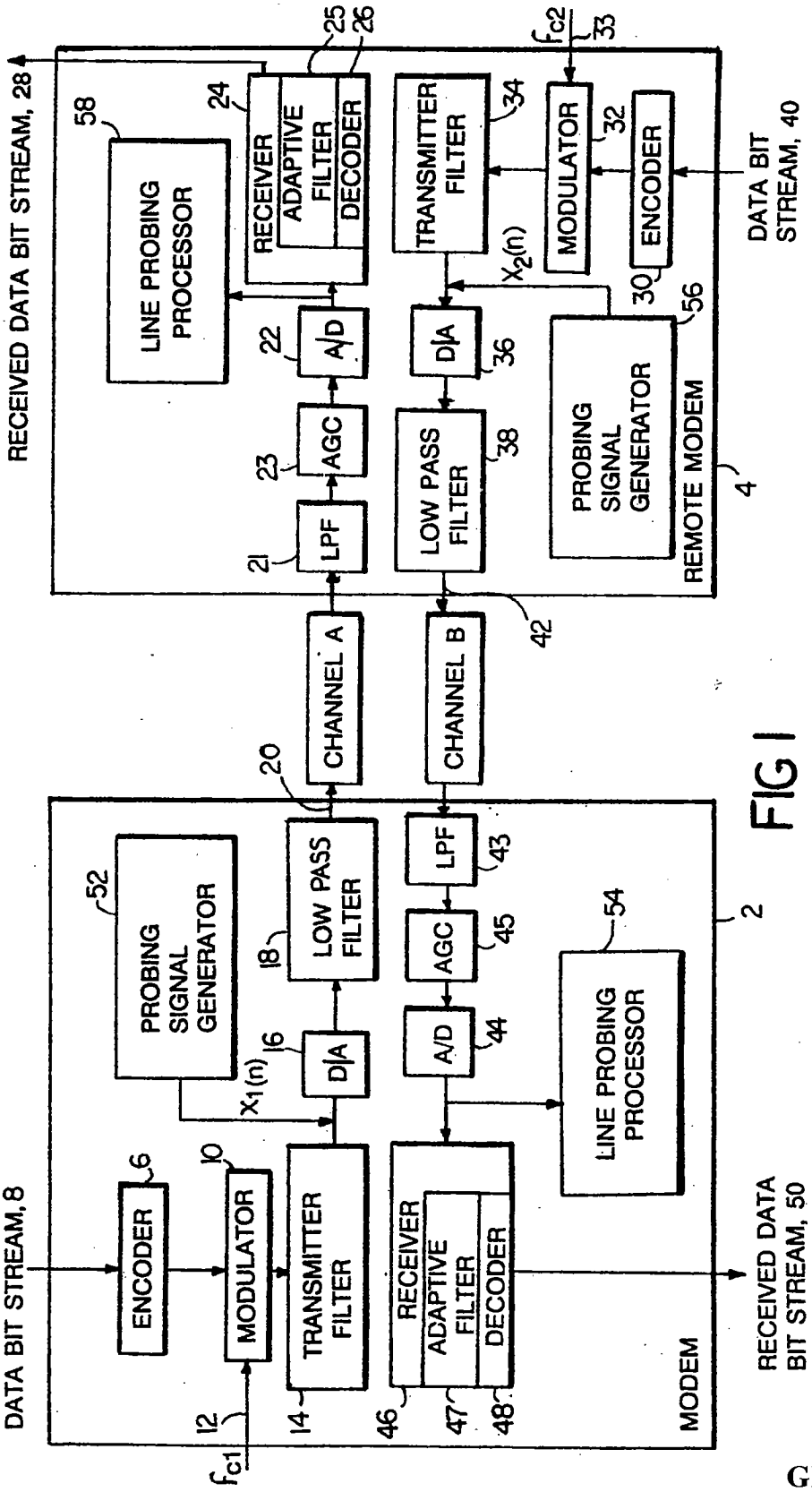


FIG 1

GE 000003

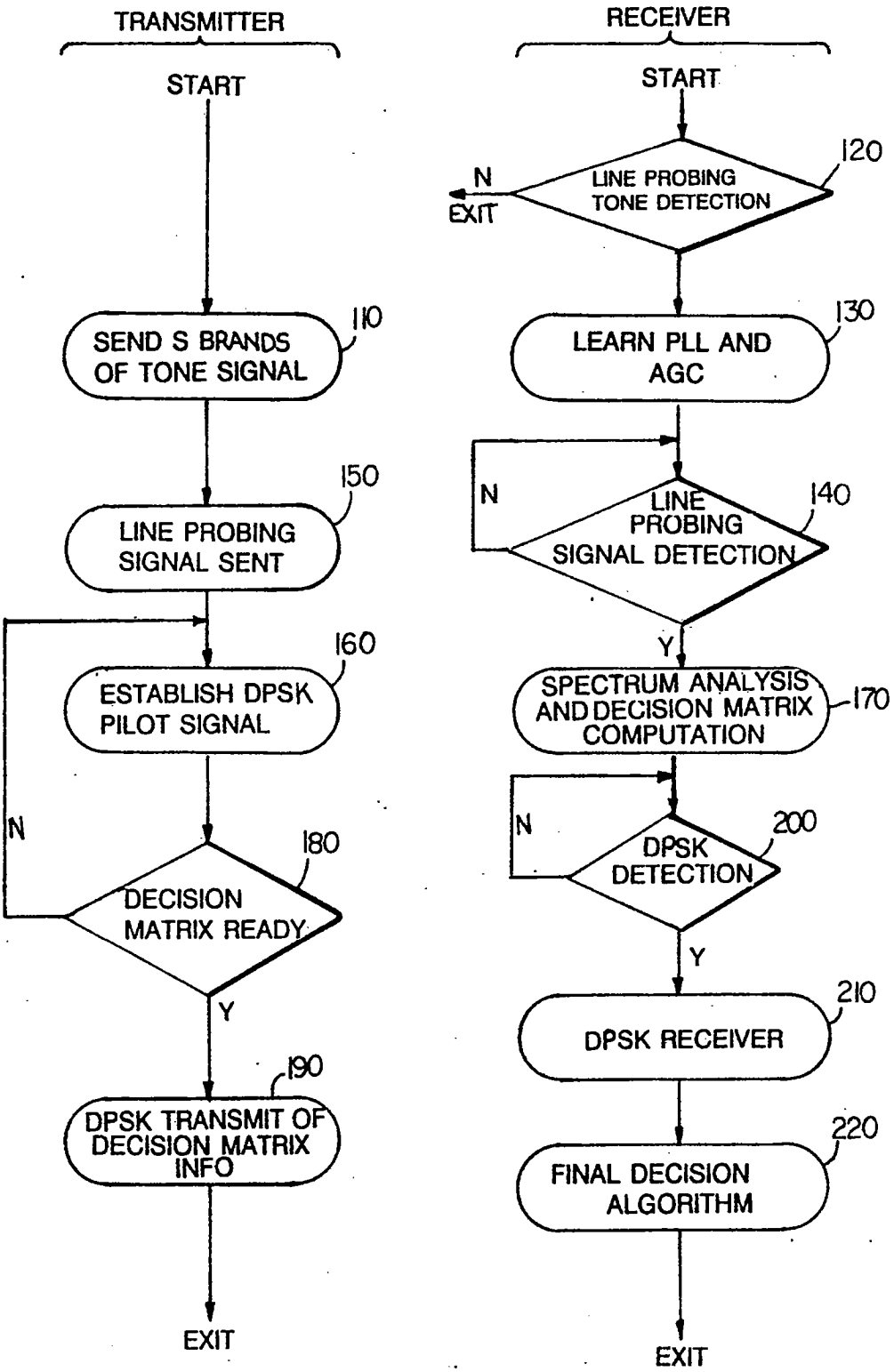


FIG. 2

GE 000004

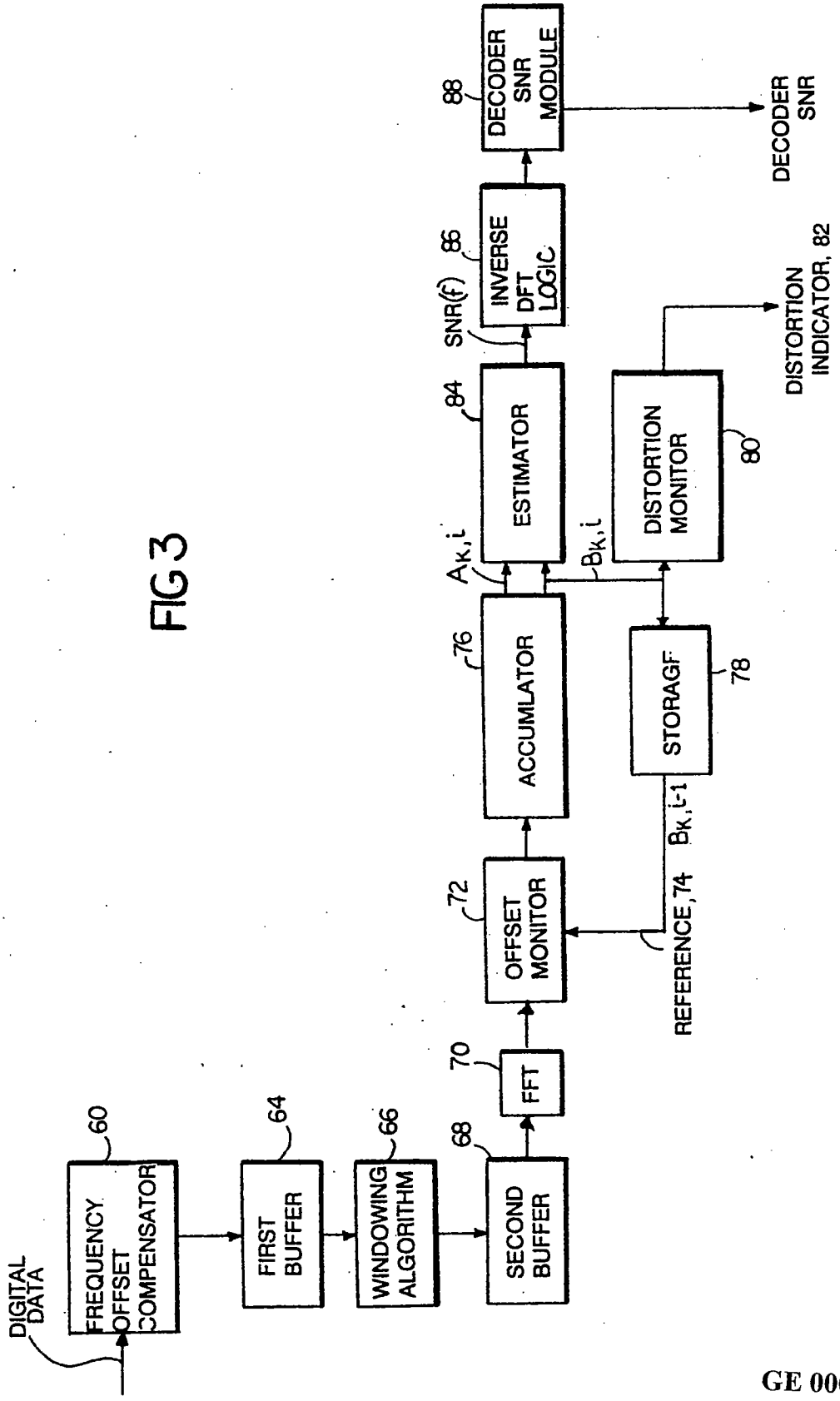


FIG 3

GE 000005

LINE PROBING MODEM

BACKGROUND OF THE INVENTION

This invention relates to data communication equipment or modems.

Modems are devices which employ digital modulation techniques to transmit binary data over analog band limited communication channels. High-speed modems commonly use linear modulation schemes such as quadrature amplitude modulation (QAM).

In linear modulation systems, binary information is collected in groups of M L bits (M is dimensionality and L is the bits/baud which may be fractional) and the resulting sequence is mapped into a sequence of complex-valued signal points, using some coding scheme. The complex sequence is filtered by a shaping filter to limit its bandwidth, and the real and imaginary components of the filtered signal points are used to amplitude modulate the quadrature components of a sinusoidal carrier of some frequency f_c. If the bit rate is R b/s, then Q=R/L is the baud rate of the linear modulation system. The baud rate represents the minimum bandwidth required to transmit the modem signal without introducing distortion. (The actual bandwidth of the shaping filter may be larger, but it is typically proportional to the baud rate.) The baud rate and the carrier frequency together determine the transmission band.

The bandwidth efficiency of a linear modulation system is measured by L, the number of bits it transmits per baud. For fixed rate R, increasing L reduces the baud rate and thus the required bandwidth. However, increasing L also reduces the noise tolerance of the system. Therefore, for a given channel characteristic, there is an optimum tradeoff between the baud rate and the number of bits transmitted per baud.

On channels with a rectangular or brickwall-like spectrum and white noise, the baud rate must be chosen approximately equal to the channel bandwidth. On the other hand, if the channel spectrum shows gradual attenuation, it may be preferable to choose the baud rate large enough such that portions of the attenuated regions are included in the transmission band. A large baud rate results in increased distortion, however, an equalizer in the receiver can compensate for the distortion and the noise enhancement caused by equalization may be more than offset by the improved noise tolerance obtained with a smaller L.

In most commercial high-speed voiceband modems that are available today, the baud rate and carrier frequency and thus the transmission band is often fixed; e.g., Q=2400 Hz and f_c=1800 Hz. Recently, modems were introduced which offer multiple but manually selectable carrier frequencies. In either case, since channel characteristics show considerable variation between different lines or connections, with such modems it is difficult to achieve the best possible performance on all possible lines.

SUMMARY OF THE INVENTION

In general, in one aspect, the invention is a modem for receiving data sent from a remote device over a communication channel by using a single carrier modulated signal. The modem includes a receiver for receiving the modulated signal and for receiving a line probing signal sent by the remote device over the channel, the receiver being capable of receiving the modulated signal over any one of a plurality of frequency bands; a line probing processor for measuring characteristics of the channel based upon the received line probing signal; and a selector for selecting one of the plurality of frequency bands, said selection being based upon the measured characteristics of the channel, said selected frequency band to be used for receiving the modulated signal from the remote device.

In preferred embodiments, the measured characteristics include a frequency response of the channel and/or a signal-to-noise ratio of the channel measured at more than one frequency. The receiver includes an adaptive filter (which may implement trellis precoding) for providing a desired overall impulse response for the channel and at least some of the measured characteristics take into account the adaptive filter. And, the line probing signal is a substantially periodic signal.

Preferred embodiments also include the following features. The line probing processor includes a spectrum analyzer for generating discrete spectral representations of the received line probing signal; a module for estimating a frequency response for the channel based upon the discrete spectral representations of the received line probing signal, the frequency response being estimated at more than one frequency; and a module for estimating a power spectral density of channel noise based upon the discrete spectral representations of the received line probing signal. Also, the modulated signal is a linearly modulated signal (such as a quadrature amplitude modulated signal) and each one of said plurality of frequency bands is characterized by a corresponding baud rate and carrier frequency. The noise estimating module also estimates a power spectrum of the channel response based upon the discrete spectral representations of the received line probing signal and then computes a signal-to-noise ratio corresponding to the channel based upon both the power spectral density of channel noise and the power spectrum of the channel response. The noise estimating module performs weighted periodogram averaging to estimate the power spectral density of channel noise based upon the discrete spectral representations of the received line probing signal and also concurrently estimates the power spectrum of the channel response and the power spectral density of channel noise from the same received line probing signal. The modem also includes a transmitter for transmitting information based upon the measured characteristics to the remote device so that the remote device may identify one of said plurality of frequency bands based upon said transmitted information and then communicate said identified band to the receiver and wherein the selector selects the identified band as said selected band. If the received line probing signal may include an impairment (e.g. frequency offset and/or low frequency phase jitter), the line probing processor also includes an offset monitor for reducing effects of said impairment on the discrete representation of the received signal prior its being used to determine the power spectral density of channel noise. The offset monitor reduces the effects of said impairment by first estimating said impairment and by then rotating the discrete representation corresponding to a current period of the received line probing signal by an amount determined by the estimate of said impairment. The offset monitor estimates said impairment by comparing the discrete spectral representation corresponding to the current period of the received line probing signal to a reference signal derived from the discrete spectral

representations corresponding to at least one previous period of the received line probing signal. Also, the discrete spectral representations of the received line probing signal are M-point Discrete Fourier Transforms.

Preferred embodiments also include these additional features. The receiver is capable of receiving the modulated signal at any one of a plurality of bit rates and the modem further includes logic for selecting one of the plurality of different bit rates based upon the measured characteristics of the receiver channel, said selected bit rate to be used for receiving the modulated signal from the remote device. The line probing processor includes a spectrum analyzer for generating discrete spectral representations of the received line probing signal; and a module for computing a nonlinear distortion indicator based upon the discrete spectral representations of the received line probing signal. The receiver also includes a monitor circuit for measuring a power level of the received line probing signal and the measured characteristics includes a quantity derived from the received power level.

In general, in another aspect, the invention is a modem for transmitting data to a remote device over a communication channel by using a single carrier modulated signal. The modem includes a signal generator for generating a line probing signal; a transmitter for transmitting the modulated signal and for transmitting the line probing signal to the remote device over the channel, the transmitter being capable of transmitting the modulated signal over any one of a plurality of frequency bands; a receiver for receiving characteristics of the channel from the remote device, the characteristics being derived by the remote device from the transmitted line probing signal; and a selector for selecting one of the plurality of frequency bands, said selection being based upon the measured channel characteristics, the selected frequency band to be used for transmitting the modulated signal to the remote device.

In general, in yet another aspect, the invention is a modem for receiving data sent from a remote device over a communication channel by using a single carrier modulated signal. The modem includes a receiver for receiving the modulated signal and for receiving a line probing signal sent by the remote device over the channel, the receiver being capable of receiving the modulated signal at any one of a plurality of bit rates; a line probing processor for measuring characteristics of the channel based upon the received line probing signal; and a selector for selecting one of the plurality of bit rates, said selection being based upon the measured characteristics of the receiver channel, the selected bit rate to be used for receiving the modulated signal from the remote device.

The invention determines the best transmission band and maximum bit rate for the modem based upon an offline measurement of the characteristics of the particular channel to which the modem is connected. Thus, in comparison to other conventional modems which use a single carrier frequency modulation scheme, the invention makes better use of the available frequency band of the channel and does so from the beginning of data transmission. Moreover, for modems which utilize an adaptive rate system to establish and maintain optimum performance during the course of data communications, the invention provides an efficient way to initialize the adaptive rate system.

Furthermore, when using a QAM system, the invention achieves close to optimal utilization of the maximum theoretical capacity of the channel.

Other advantages and features will become apparent from the following description of the preferred embodiment, and from the claims.

DESCRIPTION OF THE PREFERRED EMBODIMENT

We first briefly describe the drawings.

FIG. **1** is a block diagram of a communication system which embodies the invention;

FIG. **2** is a flow chart depicting the operation of the line probing shown in FIG. **1**; and

FIG. **3** is a block diagram of the portion of the modem which implements the spectrum analysis and the decision matrix computation step shown in FIG. **2**.

STRUCTURE AND OPERATION

Referring to FIG. **1**, a local modem **2**, which is of a fourwire type, transmits information to a remote modem **4** over a channel A and receives information sent by the remote device **4** over a channel B. In local modem **2**, an encoder **6** receives a data bit stream **8** and encodes the bits according to some coding scheme at a baud rate Q_1 selected from a set of available baud rates. Local modem **2** sends L_1 bits/baud where L_1 is selected based on a set of available bit rates. A modulator **10**, using a carrier signal **12** of a frequency f_{c1} selected from a set of available carrier frequencies, modulates the output of encoder **6** and a transmit filter **14** produces pulse shaping to control the bandwidth of the transmit signal, all in accordance with some single carrier modulation scheme, e.g. quadrature amplitude modulation (QAM). Next, a digital-to-analog (D/A) converter **16** and a low pas filter **18** convert the digital transmit signal to an analog signal **20** which is transmitted over channel A to remote modem **4**.

In remote modem **4**, the received signal passes through a lowpass filter **21**, an automatic gain control (AGC) circuit **23**, an analog-to-digital (A/D) converter **22** and then a receiver **24**, which includes an adaptive filter **25** followed by a decoder **26**. Adaptive filter **25** provides a desired overall impulse response for decoder **26** which decodes the received signal according to the particular coding scheme used by local modem **2** to obtain an estimate of the transmitted data bit stream **8**.

Remote modem **4** also includes an encoder **30**, operating at a baud rate Q_2 selected from a set of available baud rates, a modulator **32** with a carrier signal **33** of frequency f_{c2}, selected from a set of available carrier frequencies, a transmitter filter **34**, a digital-to-analog converter **36** and a low pass filter **38** which convert a data bit stream **40** into an analog signal **42** for transmission over channel B. Remote modem **4** sends L_2 bits per baud selected based on a set of available bit rates. Likewise, local modem **2** includes a low-pass filter **43**, an automatic gain control circuit **45**, an analog-to-digital converter **44** and a receiver **46**. Similarly, receiver **46** includes an adaptive filter **47** which produces a desired overall impulse response for a decoder **48** which decodes the signal received over channel B to generate an estimate **50** of the data bit stream **8** transmitted by remote modem **4**.

Local modem **2** includes a line probing signal generator **52**, which generates a special probing signal sequence $x_1(n)$, and a line probing processor **54** which measures the quality of channel B. Likewise, remote

modem 4 includes a corresponding line probing signal generator 56, which generates a probing signal sequence $x_2(n)$, and a line probing processor 58 which measures the quality of channel A.

In general, local modem 2 sends its probing signal sequence $x_1(n)$ to line probing processor 58 of the remote modem 4, which uses the corresponding received signal sequence to compute the signal-to-noise ratio (SNR) for channel A as a function of frequency, i.e. $SNR_A(f)$. Then, for each combination of baud rate and carrier frequency available to it, remote modem 4 computes a corresponding decoder signal-to-noise ratio (which shall be defined shortly). For each baud rate, the carrier frequency which yields the best performance (i.e., as will be explained later, the highest decoder SNR) is saved along with a value representing the performance. These decisions are sent to local modem 2. Similarly, local modem 2 makes decisions based on a similar line measurement process and, in turn, sends its decisions to remote modem 4. Both modems then use the combined information to select the transmission bands (characterized by Q_1, Q_2, f_{c1}, and f_{c2}) and the transmission rates (determined by L_1 and L_2) to be used during subsequent data communications.

The probing sequences are periodic signals selected to fully and uniformly stimulate the entire channel over the spectrum of frequencies which may be useful for data communication. One such sequence consists of a group of equal amplitude tones which are evenly spaced within the frequency band of interest, namely, 100 to 3600 Hz. The frequency separation between the tones determines the frequency resolution of the resulting SNR measurements. It is desirable to select the phases of these tones so as to yield a relatively small peak-to-average ratio for the transmitted signal thereby reducing the possibility of driving the channel beyond its region of linear operation. The following is an example of one such probing sequence which satisfies these criteria:

$$x(n) = A \sum_{k_1}^{k_2} \cos(2\pi k f_\Delta n T_s + \theta_k),\ n = 0, 1, \ldots, P-1 \quad (1)$$

$$\theta_k = \pi(k-k_1)^2/(k_2-k_1) \quad (2)$$

where

A is a scaling constant;

n is a sampling interval index;

f_Δ is the frequency resolution;

k is a frequency interval index;

k_1 specifies the lowest frequency index included in the sequence;

k_2 specifies the highest frequency index included in the sequence;

T_s equals $1/f_s$, where f_s is the sampling rate; and

P equals f_s/f_Δ, the number of samples in one period of the line probing signal.

In the embodiment described herein, the sampling rate is 9600 Hz, P equals 256, the frequency resolution f_{66} is 37.5 Hz, k_1 equals 3 and k_2 equals 96 (i.e., covering a frequency range from 112.5 to 3600 Hz).

Line probing processors 54 and 58 employ the Fast Fourier Transform (FFT) technique to compute SNR(f) for their respective channels. They determine SNR(f) by measuring the frequency response, H(f), and the noise power spectral density, $\Phi(f)$, of the channel at the discrete frequencies excited by the probing signal, i.e. kf_Δ, where $k=k_1, \ldots, k_2$. Then, the processors 54 and 58 compute SNR(f) by using the following well-known relationship:

$$SNR(f) = |H(f)|^2/\Phi(f) \quad (3)$$

Before describing the steps of the measurement algorithm in detail, an explanation of the underlying rationale will be given.

In general, the real-valued received sequence sampled at times $(iP+n)T_s$ can be written in the form:

$$r(i,n) = x(n)*h(n) + w(i,n) \quad (4)$$

$$= y(n) + w(i,n)\ i = 0, 1, \ldots, N-1; \quad (5)$$

$$n = 0, 1, \ldots, P-1$$

where * signifies convolution, x(n) is a transmitted periodic probing signal, h(n) is the sampled channel impulse response, w(i,n) is a potentially colored noise sequence with a power spectral density of $\Phi(f)$, N is the number of observation periods, and i is an index for observation periods.

Since the probing signal x(n) has a flat spectrum within the frequency band of interest, the noiseless channel output y(n) has the following power spectrum:

$$|\Psi(kf_\Delta)|^2 = |H(kf_\Delta)|^2$$

where $H(kf_\Delta)$ is the Discrete Fourier Transform (DFT) of h(n) and $\Psi(kf_\Delta)$ is the DFT of y(n).

Estimates of both $H(kf_\Delta)$ and $\Phi(kf_\Delta)$ may be readily obtained from a P-point DFT of the received segment r(i,n) n=0,1, . . . , P-1, which is designated hereinafter as $R_i(kf_\Delta)$, i=0,1, . . . , N-1. If $\Psi(kf_\Delta)$ were precisely known, then an estimate of the noise spectrum $\Phi(kf_\Delta)$ could be obtained from the following periodogram average:

$$\Phi(kf_\Delta) \approx (1/N) \sum_{i=0}^{N-1} \left| R_i(kf_\Delta) - \Psi(kf_\Delta) \right|^2;\ k_1 \leqq k \leqq k_2. \quad (6)$$

It can be shown that the periodogram averaging yields an asymptotically unbiased and consistent estimate of the noise spectrum, $\Phi(kf_\Delta)$. That means, as the number of observation periods increases, the mean and variance of the error tend to zero.

Although $\Psi(kf_\Delta)$ is unknown, it may be estimated by using the following DFT averaging:

$$\Psi(kf_\Delta) \approx (1/N) \sum_{i=0}^{N-1} R_i(kf_\Delta);\ k_1 \leqq k \leqq k_2. \quad (7)$$

After substituting Eq. 7 into Eq. 6, with straightforward manipulations, Eq. 6 can be written in the following form:

$$N\Phi(kf_\Delta) \approx \Sigma_i |R_i(kf_\Delta)|^2 - |(1/\sqrt{N})\Sigma_i R_i(kf_\Delta)|^2 \quad (8)$$

With the substitutions

$$A_k = \Sigma_i |R_i(kf_{66})|^2;\ k_1 \leqq k \leqq k_2 \quad (9)$$

$$B_k = (1/\sqrt{N})\Sigma_i R_i(kf_{66});\ k_1 \leqq k \leqq k_2 \quad (10)$$

this further reduces to:

$$\Phi(kf_{66}) \approx (A_k - B_k^2)/N;\ k_1 \leqq k \leqq k_2. \quad (11)$$

Note that:

$$|H(kf_\Delta)|^2 \approx B_k^2;\ k_1 \leqq k \leqq k_2. \quad (12)$$

The line probing processors **54** and **58** use the above equations to simultaneously estimate the noise spectral density $\Psi(kf_\Delta)$ and the channel frequency response $H(kf_\Delta)$.

Every observation period, the algorithms accumulate and store $\Sigma_i|R_i(kf_\Delta)|^2$ and $(1/\sqrt{N})\Sigma_i R_i(kf_\Delta)$. After N observation periods, the results equal A_k and B_k, $k=k_1, \ldots, k_2$, respectively.

The estimate of $\Phi(kf_\Delta)$ at any given frequency kf_Δ may be affected by the spectral energy density at other frequencies and thus may be "biased". When computing the DFT's from the received signal r(i,n) with an FFT technique, the line probing processors **54** and **58** use windowing, a known spectrum analysis technique for improving the performance of simple periodogram averaging. In the embodiment described herein, a Hanning window of length 2P is used. The Hanning window has a raised cosine shape with a 100% roll-off. To reduce the calculation time associated with using a window of length 2P, two successive periods of received data are overlapped. Of course, a window of duration longer than 2P may be used to improve accuracy, however, this would increase the amount of computations required for completing the FFT calculations.

The received signal r(i,n) may also have a frequency offset that can substantially degrade the accuracy of the estimation of $R_i(kf_\Delta)$ and, in turn, the noise spectrum. Although, windowing reduces effects of the frequency offset, additional steps are taken to reduce it even further. In the noise-free situation, the presence of frequency offset causes the DFT for the current observation period to differ from the DFT for the preceding period by a constant phase factor. That is:

$$R_{i+1}(kf_\Delta)/R_i(kf_\Delta) = \text{phase factor} = \exp[j2\pi f_0 PT_s] \quad (13)$$

where f_0 is the frequency offset. This relationship is used to estimate the phase factor. Then, the estimated phase factor is used to rotate the DFT's to cancel the effects of the frequency offset.

In the embodiment described herein, line probing processors **54** and **58** use the accumulating estimates of B_k, $k=k_1, \ldots, k_2$, (which shall be designated as $B_{k,i}$, where i indicates the observation period) rather than the DFT from the previous period, to achieve basically the same results. That is, after the initial period of accumulation, the newly computed periodograms $R_i(kf_\Delta)$ are compared with the corresponding $B_{k,i-1}$ by taking the inner product:

$$E = \Sigma_k B_{k,i-1} R_i^*(kf_\Delta) \quad (14)$$

where * is the complex conjugate and the summation is over $k_1 \leqq k \leqq k_2$. The quantity E, a complex number, is normalized by using a polynomial approximation of the function $1/\sqrt{x}$ where x equals $[|\text{Re}[E]|^2 + |\text{Im}[E]|^2]$ and then it is used to rotate $R_i(kf_\Delta)$ before $R_i(kf_\Delta)$ is added to $B_{k,i-1}$ to produce $B_{k,i}$, a new estimate of B_k.

In addition to SNR(f), an important source of distortion for data transmission is non-linear distortion (NLD). NLD causes the energy in the transmitted frequency components to be spread over other frequencies. Although there are standard techniques for measuring NLD on telephone lines, it is desirable to obtain a rough estimate of NLD by using the line probing signals. Thus, to measure NLD, the line probing signal is slightly modified from the one described above. This is done by omitting some preselected frequency lines. Line probing processors **54** and **58** then measure the strength of the received line probing signals at these omitted frequencies and average those measurements to arrive at a rough estimate of NLD. As will be described later, the estimate of NLD is then taken into account in estimating the maximum achievable bit rates for the modems.

To account for the missing frequencies, line probing processors **54** and **58** estimate the missing values of $SNR(kf_\Delta)$ by averaging the values of $SNR(kf_\Delta)$ corresponding to frequency lines in the vicinity of the omitted frequency lines. The location of the omitted frequency lines are selected so that they are common to all transmission bands available to the modem and lie near the mid-range of such bands, where SNR(f) is likely to be relatively uniform. By selecting the omitted frequencies in this manner, the error caused by this approximation is kept small.

After N observation periods have elapsed and the estimates of A_k and B_k have been accumulated for $k_1 \leqq k \leqq k_2$, processors **54** and **58** compute $SNR(kf_\Delta)$ for the corresponding channels using Eqs. 11, 12 and 3, above. The computed $SNR(kf_\Delta)$ is then used to determine for each baud rate the carrier frequency which yields the best receiver performance. The way this is determined will now be described.

The decoder in each modem operates on a properly equalized signal, i.e. one which has passed through the receiver's adaptive filter. Thus, it is generally the SNR at the output of the adaptive filter, i.e. the decoder SNR, that is most relevant to the performance of the receiver. The decoder SNR is related to $SNR(kf_\Delta)$, and the precise relationship depends upon the type of adaptive filter used in the receiver.

In the described embodiment, modems **2** and **4** are equipped with a trellis precoding equalization system such as the scheme disclosed in the U.S. patent application entitled "Trellis Precoding for Modulation Systems, "by Eyuboglu and Forney, filed on an even date herewith and hereby incorporated by reference. In trellis precoding, each receiver **24** and **46** includes a fractionally-spaced minimum-mean-squared error linear equalizer whose output is sampled at the baud rate, followed by a linear prediction filter responsible for whitening the residual error sequence at the output of the linear equalizer. If $\{x_n'\}$ is the complex sequence transmitted by the trellis precoder, then the received sequence at the output of the prediction filter can approximately be written in the form

$$\{r'_n\} = \{x_n'\} * \{h'_n\} + \{w_n\}, \quad (15)$$

where $\{h'_n\}$ is a causal (i.e., $h'_{nb}=0$, $n>0$) overall impulse response and $\{w_n\}$ is a white error sequence of some variance σ^2. Here, it may be assumed that the filters are scaled such that $h_0=1$. Then, under the assumption that the error signal can be modeled as Gaussian and neglecting other small effects, it is known that the performance of the trellis precoder is given by

$$P_e \approx K\, Q(d_{min}/2\sigma) \quad (16)$$

where K is a constant, $Q(\alpha)$ is the Gaussian tail function given by

$$Q(\alpha) = (1/\sqrt{2\pi}) \int_{\infty}^{\alpha} \exp\{-a^2/2\}\, da \quad (17)$$

and d_{min} is typically taken as the minimum distance between L allowable channel output sequences. The quantities K and d_{min} depend on the trellis code that is used in conjunction with the trellis precoder. Stated approximately, d_{min} decreases by a factor of $\sqrt{2}$ for every increment in L, the number of bits per baud, assuming that the average power of the precoder output is kept constant. The sampled decoder SNR may be defined as $d^2_{min}/2\sigma^2$. The baud rate or carrier frequency affect the decoder SNR only through the noise variance σ^2.

To determine the relationship between σ^2 and SNR(f), first note that at the equalizer output, after demodulation, the noise spectrum is given by

$$S_n(f) = \Phi(f)/|H(f)|^2, \quad |f - f_c| < Q/2, \quad (18)$$

where Q equals the baud rate and it is assumed that the transmitted signal has zero excess-bandwidth. (Typically, highspeed modems use 10–12% excess bandwidth; however, experiments have shown that this has only a small effect on performance). Now, the autocorrelation sequence of the noise sequence can be computed as

$$g_n = T \int_{|f - f_c| < Q/2} \Phi(f)/|H(f)|^2\, e^{j2\pi f n/Q}\, df, \quad n = 1, 2, \ldots \quad (19)$$

Since the spectra are measured at discrete frequencies kf_Δ, g_n can be approximated as follows:

$$g_n = a \sum \Phi(kf_\Delta)/|H(kf_\Delta)|^2 \exp(j2\pi kf_\Delta n/Q), \quad n = 1,2, \ldots \quad (20)$$

$$k_1(Q,f_c) < k < k_2(Q,f_c)$$

where a is some normalization factor and $k_1(Q,f_c)$ and $k_2(Q,f_c)$ are the frequency indices corresponding to the bandedges assuming the baud rate Q and carrier frequency f_c. Once $\{g_n\}$ are determined, σ^2 can be computed using well-known formulas for linear prediction. For example, if it can be assumed that the noise sequence is a first-order autoregressive (AR) process as described by Eyuboglu in "Detection of Severely Distorted Signals Using Decision Feedback Noise Prediction with Interleaving" IEEE Trans. on Communications, April, 1988, then, σ^2 is given by

$$\sigma^2 = g_0 - |g_1|^2/g_0. \quad (21)$$

Thus, by using Eqs. (20) and (21), line probing processors **54** and **58** can determine for each baud rate Q, the carrier frequency $f_c(Q)$ which yields the smallest noise power $\sigma^2(f_c(Q))$.

The computation of decoder SNR described above can be extended to higher order AR models using well known formulas for the minimum-mean-square error prediction error as described in Jayant and Noll, "Digital Coding of Waveforms" Prentice-Hall, 1984.

The description given above assumes that the fractionally-spaced linear equalizer has a sufficiently large span to reduce the effects of phase distortion to a negligible level. In highspeed modems that is often a reasonable assumption. In applications where this condition may not be satisfied, the effect of phase distortion has to be taken into account. Furthermore, if the assumption of a first-order AR model does not hold, then the residual noise sequence may be correlated and its effect on performance may have to be taken into consideration.

Having described the nature of the computations performed by processors **54** and **58**, the steps of the measurement algorithm, shown in FIG. **2**, will now be described in more detail. Local modem **2** starts the line probing process by transmitting a line probing tone signal at some fixed frequency for a fixed number of bauds (designated in FIG. **2** as S) over channel A (step **110**) while at the same time, remote modem **4** monitors channel A to detect the tone signal (step **120**).

After detecting the line probing tone, remote modem **4** initiates a period of tone training (step **130**). During this period, it uses a phase-locked loop (PLL) to learn the frequency offset in the incoming tone and at the same time it adjusts its AGC setting to achieve a desired signal level prior to A/D conversion. After a fixed amount of time, the receiver freezes both its AGC setting and its PLL and then switches to a transition detection state to detect the arrival of the wideband line probing signal transmitted from the local modem **2** (step **140**).

In the meantime, the transmitter in modem **2** continues to transmit its tone for at least S bauds and until it receives a reply tone from remote modem **4**. Then, processor **54** causes probing signal generator **52** to generate the above-describe special probing signal and transmits it to remote modem **4** for at least N periods (step **150**). After receiving N periods of the probing signal from remote modem **4**, modem **2** switches to a communication mode for sending line probing measurement results to remote modem **4** (step **160**).

Since the line probing algorithm implemented by remote modem **4** is the same as the line probing algorithm of modem **2**, the sequence of events in both modems **2** and **4** and their timing is also basically the same. Thus, remote modem **4** sends its tone signal, generates its probing signal for transmission to modem **2**, and enters a corresponding communication mode at about the same times as these events occur in modem **2**.

As soon as remote modem **4** detects the probing signal on channel A, processor **58** begins a spectrum analysis of the received probing signal (step **170**). First, line probing processor **58** measures the channel and noise spectra and from these computes SNR(f) and then the decoder SNRs.

The elements of the modem which perform the spectrum analysis and the decision matrix computation of step **170** are shown in FIG. **3**. After being converted to a digital signal, the received real-valued probing signal passes through a frequency offset compensator **60** which multiplies it by a complex-valued rotation factor which was derived from the frequency offset estimate obtained during the initial tone training described above. A first buffer **64** temporarily stores the rotated digital signal for subsequent processing.

After buffer **64** has received data corresponding to two periods of the line probing signal, a windowing algorithm **66** applies a Hanning window to the two periods of rotated data stored in buffer **64** to produce a frame of windowed data which consists of 2P complex-valued samples. These are then stored in a second buffer

68. Next, FFT algorithm 70 computes a P-point DFT from the stored frame of windowed data.

After each new period of data is received and stored in first buffer 64, windowing algorithm 66 uses the stored data, along with the data from the preceding period, to compute a new frame of windowed data. FFT algorithm 70 then computes a new P-point DFT using the new frame of windowed data. In other words, for each period of the received probing signal, a new P-point DFT is generated from the two most recent periods of data. Thus, one period of data is used to compute the DFT for two successive periods.

The P-point DFT's from FFT algorithm 70 are passed to a frequency offset monitor 72, which first estimates and then reduces any uncancelled frequency offset which may be present. Offset monitor 72 estimates the amount of uncancelled frequency offset by comparing each computed DFT against a reference 74 which corresponds to accumulated DFT's from previous observation periods. Offset monitor 72 then rotates the elements of the DFT by an amount that corresponds to the estimated uncancelled frequency offset for that DFT, thereby generating the rotated DFT, $R_i'(kf_\Delta)$.

Next, an accumulator 76 receives these rotated DFT's and uses them to generate the quantities A_k and B_k as follows. Initially, accumulator 76 sets both $A_{k,0}$ and $B_{k,0}$ to zero. During the i^{th} period, when $R_i'(kf_\Delta)$ has been compiled, for each k in the range $k_1 \leqq k \leqq k_2$, accumulator 76 computes the squared magnitude of $R_i'(kf_\Delta)$ and adds the result to the stored value for $A_{k,i-1}$ to generate $A_{k,i}$. Accumulator 76 also divides the $R_i'(kf_\Delta)$ by $\sqrt{N}$ and adds this result to the stored value for $B_{k,i-1}$ to generate $B_{k,i}$. $A_{k,i}$ and $B_{k,i}$ are accumulated in this manner for N observation periods. (Note that $A_{k,i}$'s are real numbers, whereas $B_{k,i}$'s are complex numbers.)

The $B_{k,i}$'s generated during each observation period are stored in storage element 78 for use as a reference signal 76 during the next observation period. That is, for the i^{th} observation period, reference signal 76 is equal to $B_{k,i-1}$ computed during the previous observation period. Offset monitor 72 uses reference signal 76 to compute Eq. 14 described earlier. That is, offset monitor 72 calculates the complex conjugate of the current P-point DFT from FFT algorithm 70, i.e. $R_i^*(kf_\Delta)$, and then computes the inner product of $B_{k,i-1}$ and $R_i^*(kf_\Delta)$. The inner product is then normalized to arrive at an estimate of the phase offset. It should be noted that by using this approach, the modem can compensate for any residual frequency offset as well as track small amounts of low frequency phase jitter.

Since preselected frequencies were omitted from the line probing signal, the values of B_k at the locations of the omitted frequencies provide a measure of the non-linear distortion associated with the channel. Thus, after $B_{k,N}$ has been determined for N observation periods, a distortion monitor 80 estimates the non-linear distortion by squaring the amplitudes of $B_{k,N}$'s corresponding to omitted frequencies and computing the average of the resulting squared amplitudes. The average is then supplied by monitor 80 as a non-linear distortion indicator 82.

Using the $A_{k,N}$'s and $B_{k,N}$'s from accumulator 76, an estimator 84 then estimates the noise spectrum, $\Phi(kf_\Delta)$, and the channel spectrum, $|H(kf_\Delta)|^2$, for all frequencies used in the probing signal. Estimator 84 accomplishes this by first computing $|B_{k,N}|^2$ and then using Eqs. 11 and 12, described above. Using the estimates for the noise spectrum and the channel spectrum, estimator 84 then computes $SNR(kf_\Delta)$ in accordance with Eq. 3, described above. For the omitted frequency lines, estimator 84 approximates their SNR value by averaging the values of $SNR(kf_\Delta)$ over frequency lines in their vicinity.

Transform logic 86 receives the resulting values for $SNR(kf_\Delta)$ from estimator 84 and computes the inverse DFT specified by Eq. 20, above. The output of transform logic 84 is the noise autocorrelation function described earlier. Finally, a decoder SNR module 88 calculates the decoder SNR from the output of transform logic 86 according to Eq. 21.

Using the approach just described and also as part of step 170 shown in FIG. 2, remote modem 4 then makes a number of local decisions. Such local decisions help reduce the amount of information that needs to be exchanged with local modem 2. (Note that the local decision procedures to be described are the same for both modems 2 and 4. In particular and in accordance with the approach described earlier, remote modem 4 uses $SNR(kf_\Delta)$ to compute the decoder SNR's for each baud-rate/carrier-frequency combination available to it and then selects the best carrier frequency for each of the available baud rates.

The computed decoder SNR's, the non-linear distortion indicator, the signal power level of the received signal, as reflected by the AGC setting, and a user specified error performance requirement are then used to determine for each available baud-rate Q (using the best-carrier-frequency) and the maximum number of bits per baud $L_1(Q_1)$ that remote modem 4 can receive at without violating the performance requirement. To determine $L_1(Q_1)$, modem 4 uses a precomputed conversion table which is indexed on the basis of the above-identified information.

Basically, the conversion table depends upon the modem's modulation scheme, the coding gain of the coding scheme used, the way those schemes are implemented, and the error performance requirements. If trellis precoding is employed, the relationship between performance and decoder SNR is approximately described by Eq. 16 above. Nonlinear distortion and receive power level, however, modify that relationship somewhat. The actual entries in the conversion table can be derived, in part, from empirical observations and experiments in which the relationship between performance and the decoder SNR, NLD, and receive power level is measured for the particular type of modem being used.

After $L_1(Q_1)$ is obtained for all baud rates, remote modem 4 can calculate, for each of the available baud rates Q_1, the maximum bit rate $R_1(Q_1)$ it can receive from local modem 2 according to the following relation:

$$R_1(Q_1)=Q_1\times L_1(Q_1)$$

When the spectrum analysis is complete, line probing processor 58 stores the results in a decision matrix. Upon completing the entries to the decision matrix, processor 58 indicates that its matrix is ready (step 180) and remote modem 4 transmits the information contained in its matrix to local modem 2 over channel B (step 190).

In each modem, the user or the network system may specify a maximum receive bit rate, R_{max}, and a minimum receive bit rate, R_{min}. This user-specified operating range is taken into account when the modem deter-

mines the decision matrix entries. Thus, if the selected bit rate for a particular baud rate is greater than R_{max}, then modem 2 sets it to R_{max}. Whereas, if the selected bit rate for a particular baud rate is less than R_{min}, then modem 2 sets it to R_{min} and also sets a flag associated with that baud rate to indicate that the performance requirement cannot be met at that baud rate. Note that a user can force a desired bit rate by setting $R_{max}=R_{min}$=desired rate.

The user also has the option to disable some (but not all) of the available baud rates. For example, the user may wish to operate at a specific baud rate. A second flag corresponding to each of the available baud rates is set to indicate whether that baud rate is disabled.

Of course, other constraints, besides those mentioned above, may also limit the communication options available to modems 2 and 4. For example, a user may require symmetric baud rates or symmetric bit rates in both transmission directions. Such additional constraints are stored in the decision matrix of the corresponding modem and are sent to the other modem along with other relevant information.

Specifically, during step 190, the following information is sent from the remote modem 4 to local modem 2:

a) the maximum bit rate at which the remote modem can receive for each of the available baud rates;

b) the best carrier frequency to be used for each of the available baud rates;

c) a flag for each of the available baud rates indicating whether the performance requirement can be met;

d) a flag for each of the available baud rates, indicating whether that baud rate is disabled in the remote modem;

e) a flag to indicate whether symmetric bit rates are required for both directions of transmission; and

f) a flag to indicate whether symmetric baud rates are required for both directions of transmission.

Naturally, some synchronization bits to indicate to beginning of the data and parity bits for error checking may also be transmitted during this information exchange phase

Since the decision matrix information is short, it can be transmitted quickly and reliably by using a simple, low-speed, robust modulation scheme which does not require a long training procedure. In this embodiment, this is achieved by using Differential-Phase-Shift-Keying (DPSK) at 300 bps. Other reliable modulation schemes such as low-speed Frequency-Shift-Keying (FSK) may also be employed.

Processor 54 monitors channel B for the presence of a DPSK signal carrying the decision matrix for channel A from remote modem 4 (step 200). When the DPSK signal is detected on channel B, processor 54 activates a DPSK receiver in modem 2 that includes a timing recovery circuit to provide correct sampling phase and then processor 54 decodes the decision matrix from the DPSK signal (step 210).

The DPSK receiver first looks for a synchronization pattern from the received bit stream. Once the pattern is detected, the receiver decodes the subsequent bits carrying the decision matrix. At the same time, the receiver also computes a parity check. At the end of DPSK transmission, this parity is compared with the one received from the remote modem. If they do not agree, a DPSK transmission error is flagged.

After modems 2 and 4 have exchanged their decision matrix information in this manner, they have complete information about channels A and B, including the operational constraints. Modem 2 then executes a final decision algorithm to select the carrier frequencies baud rates and bit rates to be used for communication over channels A and B (step 220). Since both modems 2 and 4 have the same information, they make the same selections and a further exchange of final decisions is not required.

The final decision algorithm first checks if any one of the modems required symmetric bit rates, or symmetric baud rates. If one of modems requested a symmetric baud rate or a symmetric bit rate, the request is enforced on both modems.

More specifically, if symmetric baud rates are required, the decision algorithm checks whether there are allowable baud rates common to both modems (i.e., baud rates which both satisfy the performance requirements and are allowed). If such baud rates exist, modems select the baud rate that maximizes the smaller of the two bit rates. When there are no allowable baud rates common to both modems, the decision algorithm includes baud rates that do not satisfy the performance requirement to find a common baud rate. A possible criterion for determining the reasonable baud rate may be to use a baud rate whose carrier frequency is closest to the center of the frequency band. Since both modems use the same criterion to choose this baud rate, they should both reach the same conclusion and no confusion will occur.

On the other hand, if symmetric baud rates are not required, the decision algorithm chooses from all of local modem's allowed baud rates the receiver baud rate that maximizes local modem's receiver bit rate and it chooses from all of remote modem's allowed baud rates the transmitter baud rate that maximizes remote modem's receiver bit rate.

After the transmitter and the receiver baud rates for the two modems are finalized, the best carrier frequencies associated with these baud rates are used as the transmitter and receiver carrier frequencies. Unless a symmetric bit rate is required, the maximum bit rate for each of those baud rates is used as the transmission bit rate for the corresponding modem. If a symmetric bit rate is required, the lower of the two bit rates (i.e. the bit rate for the local modem receiver and the bit rate for the remote modem receiver) is used as the common bit rate.

The main outputs of the line probing processor are the transmitter and receiver baud rates, Q_1 and Q_2, the transmitter and receiver carrier frequencies, f_{c1} and f_{c2},the transmitter and receiver bit rates, R_1 and R_2, as well as an error code, which may indicate some unexpected error during the line probing process (such as failure in detecting the line probing signal, failure in synchronization, DPSK transmission error, etc.).

After the line probing is completed, modems go through a training at the selected baud rate and carrier frequencies and subsequently begin exchanging actual data at the selected rates.

Although the described embodiment used a four-wire type modem, it should be understood that this invention could also be carried out using a two-wire type modem. During full-duplex communication using a two-wire type modem, the received signals may, of course, include echo. For purposes of conducting the line probing measurements, it is desirable to avoid echoes in the received signal and this can easily be accomplished by having the modems conduct the line probing measurements sequentially rather than concurrently, as in the above-described embodiment.

Other embodiments may include the following features. The selection of the number of bits per baud may be based on the measurement of impairments in addition to or other than NLD and receive level. Also, in certain applications, no operational constraints may be necessary, in which case, the information exchange between the local modem and the remote device may be simplified. For example, each modem could immediately select its respective bit rate and the transmission band based upon its channel measurement and then exchange its final decision with the other modem. In addition, other exchange protocols may be used. Further, the selection of the transmission band may be based only on the measured frequency response of the channel and may not require measurement of the noise spectrum. Also, baseband data transmission may be employed instead of the passband transmission used in the described embodiment.

Other embodiments are within the following claims.

What is claimed is:

1. A modem for receiving data sent from a remote device over a communication channel by using a single carrier modulated signal, the modem comprising:

a. a receiver for receiving the modulated signal and for receiving a line probing signal sent by the remote device over the channel, the receiver being capable of receiving the modulated signal over any one of a plurality of frequency bands, said line probing signal simultaneously stimulating more than one of said plurality of frequency bands;

b. a line probing processor for measuring characteristics of the channel based upon the received line probing signal; and

c. a selector for selecting one of the plurality of frequency bands, said selection being based upon the measured characteristics of the channel, said selected frequency band to be used for receiving the modulated signal from the remote device.

2. The modem of claim 1 wherein the line probing processor comprises:

a. a spectrum analyzer for generating discrete spectral representations of the received line probing signal; and

b. a module for estimating a frequency response for the channel based upon the discrete spectral representations of the received line probing signal, the frequency response being estimated at more than one frequency.

3. The modem of claim 1 wherein the line probing processor comprises:

a. a spectrum analyzer for generating discrete spectral representations of the received line probing signal; and

b. a module for estimating a power spectral density of channel noise based upon the discrete spectral representations of the received line probing signal.

4. The modem of claims 1, 2, or 3 wherein the modulated signal is a linearly modulated signal and wherein each one of said plurality of frequency bands is characterized by a corresponding baud rate and carrier frequency, the modulated signal from the remote device being received at the corresponding baud rate associated with said selected frequency band.

5. The modem of claim 4 wherein the carrier frequency of two or more of said plurality of frequency bands are the same.

6. The modem of claim 4 wherein the linearly modulated signal is a quadrature amplitude modulated signal.

7. The modem of claim 1 wherein the measured characteristics include a frequency response of the channel.

8. The modem of claim 1 wherein the receiver includes an adaptive filter for providing a desired overall impulse response to a decoder and wherein at least some of the measured characteristics take into account the adaptive filter.

9. The modem of claim 8 wherein the adaptive filter is used in conjunction with trellis precoding.

10. The modem of claim 3 wherein the module performs weighted periodogram averaging to estimate the power spectral density of channel noise based upon the discrete spectral representations of the received line probing signal.

11. The modem of claim 1 wherein the line probing signal is a substantially periodic signal.

12. A modem for receiving data sent from a remote device over a communication channel by using a single carrier modulated signal, the modem comprising:

a. a receiver for receiving the modulated signal and for receiving a line probing signal sent by the remote device over the channel, the receiver being capable of receiving the modulated signal over any one of a plurality of frequency bands, each one of said plurality of frequency bands being characterized by a corresponding baud rate and carrier frequency;

b. a line probe processor for measuring characteristics of the channel based upon the received line probing signal; and

c. a selector for selecting one of the plurality of frequency bands, said selection being based upon the measured characteristics of the channel, the modulated signal from the remote device being received at the corresponding baud rate associated with said selected frequency band.

13. The modem of claim 12 wherein the line probing processor comprises:

a. a spectrum analyzer for generating discrete spectral representations of the received line probing signal; and

b. a module for estimating a frequency response for the channel based upon the discrete spectral representations of the received line probing signal, the frequency response being estimated at more than one frequency.

14. The modem of claim 12 wherein the line probing processor comprises:

a. a spectrum analyzer for generating discrete spectral representations of the received line probing signal; and

b. a module for estimating a power spectral density of channel noise based upon the discrete spectral representations of the received line probing signal.

15. The modem of claim 14 wherein the module performs weighted periodogram averaging to estimate the power spectral density of channel noise based upon the discrete spectral representations of the received line probing signal.

16. The modem of claim 15 wherein the module also estimates a power spectrum of the channel response based upon the discrete spectral representations of the received line probing signal and then computes a signal-to-noise ratio corresponding to the channel based upon both the power spectral density of channel noise and the power spectrum of the channel response.

17. The modem of claim 16 wherein the power spectrum of the channel response and the power spectral

density of channel noise are estimated concurrently from the same received line probing signal.

18. The modem of claim **16** wherein the receiver further comprises an adaptive filter for providing a desired overall impulse response to a decoder and the signal-to-noise ratio is determined relative to the output of the adaptive filter.

19. The modem of claim **18** wherein the adaptive filter is used in conjuction with trellis precoding.

20. The modem of claim **1** further comprising a transmitter for transmitting information based upon the measured characteristics to the remote device so that the remote device may identify one of said plurality of frequency bands based upon said transmitted information and then communicate said identified band to the receiver and wherein the selector selects the identified band as said selected band.

21. The modem of claim **12** wherein the measured characteristics include a frequency response of the channel.

22. The modem of claim **12** wherein the receiver includes an adaptive filter for providing a desired overall impulse response for the channel and wherein at least some of the measured characteristic take into account the adaptive filter.

23. The modem of claim **12** wherein the measured characteristics further include a signal-to-noise ratio of the channel measured at more than one frequency.

24. The modem of claim **12** wherein the line probing signal is a substantially periodic signal.

25. The modem of claim **11** or **24** wherein the line probing signal is of the form:

$$x(t)=A\Sigma_k\cos(2\pi kf_\Delta t+\theta_k),$$

where

x(t) represents the line probing signal;

t is a time variable;

A is a constant;

f_Δ is a frequency resolution;

θ_k is a phase angle;

k is a frequency interval index which belongs to a subset of the integers ranging from k_1 through k_2;

k_1 specifies the lowest frequency index included in the line probing signal; and

k_2 specifies the highest frequency index included in the line probing signal.

26. The modem of claim **25** wherein the phase angles θ_k of the line probing signal are selected to achieve a small peak-to-average ratio of the line probing signal.

27. The modem of claim **25** wherein the phase angles θ_k of the line probing signal are equal to:

$$\theta_k=\pi(k-k_2)^2/(k-k_1)$$

28. The modem of claim **12** wherein the modulated signal from the remote device is received at the corresponding baud rate and carrier frequency associated with said selected frequency band, and wherein at least some of the carrier frequencies associated with said plurality of frequency bands are different.

29. The modem of claim **1** wherein the measured characteristics further include a signal-to-noise ratio of the channel measured at more than one frequency.

30. The modem of claim **3** wherein received line probing signal may include an impairment (e.g. frequency offset and/or low frequency phase jitter) and wherein the line probing processor further comprises an offset monitor for reducing effects of said impairment on the discrete representation of the received signal prior its being used to determine the power spectral density of channel noise.

31. The modem of claim **30** wherein the line probing signal is substantially periodic and the discrete spectral representations are generated for each period of the received line probing signal and wherein the offset monitor reduces the effects of said impairment by first estimating said impairment and by then rotating the discrete representation corresponding to a current period of the received line probing signal by an amount determined by the estimate of said impairment.

32. The modem of claim **31** wherein the offset monitor estimates said impairment by comparing the discrete spectral representation corresponding to the current period of the received line probing signal to a reference signal derived from the discrete spectral representations corresponding to at least one previous period of the received line probing signal.

33. The modem of claim **2** or **3** wherein the discrete spectral representations of the received line probing signal are M-point Discrete Fourier Transforms.

34. The modem of claim **1** wherein the receiver is also capable of receiving the modulated signal at any one of a plurality of bit rates and wherein the modem further comprises logic for selecting one of the plurality of different bit rates based upon the measured characteristics of the receiver channel, said selected bit rate to be used for receiving the modulated signal from the remote device.

35. The modem of claim **34** wherein the line probing processor comprises:

a. a spectrum analyzer for generating discrete spectral representations of the received line probing signal; and

b. a module for computing a nonlinear distortion indicator based upon the discrete spectral representations of the received line probing signal.

36. The modem of claim **34** wherein the receiver comprises a monitor circuit for measuring a power level of the received line probing signal and wherein the measured characteristic includes a quantity derived from the received power level.

37. A modem for transmitting data to a remote device over a communication channel by using a single carrier modulated signal, the modem comprising:

a. a signal generator for generating a line probing signal;

b. a transmitter for transmitting the modulated signal and for transmitting the line probing signal to the remote device over the channel, the transmitter being capable of transmitting the modulated signal over any one of a plurality of frequency bands, said line probing signal simultaneously stimulating more than one of said plurality of frequency bands;

c. a receiver for receiving characteristics of the channel from the remote device, the characteristics being derived by the remote device from the transmitted line probing signal; and

d. a selector for selecting one of the plurality of frequency bands, said selection being based upon the measured channel characteristics, the selected frequency band to be used for transmitting the modulated signal to the remote device.

38. The modem of claim **37** wherein the modulated signal is a linearly modulated signal and wherein each

one of said plurality of frequency bands is characterized by a corresponding baud rate and carrier frequency.

39. The modem of claim **38** wherein the linearly modulated signal is a quadrature amplitude modulated signal.

40. The modem of claim **37** wherein the measured characteristics include a frequency response of the channel.

41. The modem of claim **37** wherein the measured characteristics further include a signal-to-noise ratio of the channel measured at more than one frequency.

42. The modem of claim **37** wherein the line probing signal is a substantially periodic signal.

43. The modem of claim **42** wherein the line probing signal is of the form:

$$x(t)=A\Sigma_k\cos(2\pi kf_\Delta t+\theta_k),$$

where

x(t) represents the line probing signal;

t is a time variable;

A is a constant;

f_Δ is a frequency resolution;

θ_k is a phase angle;

k is a frequency interval index which belongs to a subset of the integers ranging from k_1 through k_2;

k_1 specifies the lowest frequency index included in the line probing signal; and

k_2 specifies the highest frequency index included in the line probing signal.

44. The modem of claim **43** wherein the phase angles θ_k of the line probing signal are selected to achieve small peak-to-average ratio for the line probing signal.

45. The modem of claim **43** wherein the phase angles θ_k of the line probing signal are equal to:

$$\theta_k=\pi(k-k_2)^2/(k_2-k_1)$$

46. A modem for receiving data sent from a remote device over a communication channel by using a single carrier modulated signal, the modem comprising:

a. a receiver for receiving the modulated signal and for receiving a line probing signal sent by the remote device over the channel, the receiver being capable of receiving the modulated signal at any one of a plurality of bit rates;

b. a line probing processor for measuring characteristics of the channel based upon the received line probing signal; and

c. a selector for selecting one of the plurality of bit rates, said selection being based upon the measured characteristics of the receiver channel, the selected bit rate to be used for receiving the modulated signal from the remote device.

47. The modem of claim **46** wherein the line probing processor comprises:

a. a spectrum analyzer for generating discrete spectral representations of the received line probing signal; and

b. a module for estimating a frequency response for the channel based upon the discrete spectral representations of the received line probing signal, the frequency response being estimated at more than one frequency.

48. The modem of claims **46** wherein the line probing processor comprises:

a. a spectrum analyzer for generating discrete spectral representations of the received line probing signal; and

b. a module for estimating a power spectral density of channel noise based upon the discrete spectral representations of the received line probing signal.

49. The modem of claims **46**, **47**, or **48** wherein the modulated signal is a linearly modulated signal.

50. The modem of claim **49** wherein the measured characteristics include a frequency response of the channel.

51. The modem of claim **49** wherein the linearly modulated signal is a quadrature amplitude modulated signal.

52. The modem of claim **48** wherein the module performs weighted periodogram averaging to estimate the power spectral density of channel noise from the discrete spectral representations of the received line probing signal.

53. The modem of claim **52** wherein the module also estimates a power spectrum of the channel response based upon the discrete spectral representations of the received line probing signal and then computes a signal-to-noise ratio corresponding to the channel based upon both the power spectral density of channel noise and the power spectrum of the channel response.

54. The modem of claim **53** wherein the power spectrum and the spectral power density of channel noise are estimated concurrently from the same received line probing signal.

55. The modem of claim **48** wherein received line probing signal may include an impairment (e.g. frequency offset and/or low frequency phase jitter) and wherein the line probing processor further comprises an offset monitor for reducing effects of said impairment on the discrete representation of the received signal prior its being used to determine the power spectral density of channel noise.

56. The modem of claim **55** wherein the line probing signal is substantially periodic and a discrete spectral representation is generated for each period of the received line probing signal and wherein the offset monitor reduces the effects of said impairment by first estimating said impairment and by then rotating the discrete representation corresponding to a current period of the received line probing signal by an amount determined by the estimate of said impairment.

57. The modem of claim **56** wherein the offset monitor estimates said impairment by comparing the discrete spectral representation corresponding to the current period of the received line probing signal to a reference signal derived from the discrete spectral representations corresponding to at least one previous period of the received line probing signal.

58. The modem of claim **46** wherein the measured characteristics include a signal-to-noise ratio of the channel measured at more than one frequency.

59. The modem of claim **46** wherein the line probing signal is a substantially periodic signal.

60. The modem of claim **59** wherein the line probing signal is of the form:

$$x(t)=A\Sigma_k\cos(2\pi kf_\Delta t+\theta_k),$$

where

x(t) represents the line probing signal;

t is a time variable;

A is a constant;

f_Δ is a frequency resolution;

θ_k is a phase angle;

k is a frequency interval index which belongs to a subset of the integers ranging from k_1 through k_2;

k_1 specifies the lowest frequency index included in the line probing signal; and

k_2 specifies the highest frequency index included in the line probing signal.

61. The modem of claim **60** wherein the phase angles θ_k of the line probing signal are selected to achieve a small peak-to-average ratio for the line probing signal.

62. The modem of claim **60** wherein the phase angles θ_k of the line probing signal are equal to:

$$\theta_k=\pi(k-k_2)^2/(k_2-k_1)$$

63. The modem of claim **46** wherein the receiver further comprises an adaptive filter for providing a desired overall impulse response to a decoder and wherein at least some of the measured characteristics take into account the adaptive filter.

64. The modem of claim **63** wherein the adaptive filter is used in conjuction with trellis precoding.

65. The modem of claim **46** further comprising a transmitter for transmitting information based upon the measured characteristics to the remote device so that the remote device may identify one of said plurality of bit rates based upon said information and then communicate said identified bit rate to the receiver and wherein the selector selects the identified bit rate as said selected bit rate.

66. The modem of claim **46** wherein the line probing processor comprises:

a. a spectrum analyzer for generating discrete spectral representations of the received line probing signal; and

b. a module for computing a nonlinear distortion indicator based upon the discrete spectral representations of the received line probing signal.

67. The modem of claim **46** wherein the receiver comprises a monitor circuit for measuring a power level of the received line probing signal and wherein the measured characteristics includes a quantity derived from the received power level.

68. In a system in which a local modem receives data sent by a remote device over a receiver channel in the form of a first single carrier modulated signal, the modem being capable of receiving the first modulated signal over any one of a first plurality of frequency bands, a method for establishing communication conditions comprising the steps of:

a. sending a first line probing signal from the remote device to the local modem over the receiver channel, said line probing signal simultaneously stimulating more than one of said plurality of frequency bands;

b. receiving the first line probing signal in the local modem;

c. measuring characteristics of the receiver channel based upon the received first line probing signal; and

d. selecting one of the first plurality of frequency bands based upon the measured characteristics of the receiver channel, said selected one of the first plurality of frequency bands to be used for receiving the first modulated signal from the remote device.

69. The method of claim **68** wherein the modem transmits data to the remote device over a transmitter channel by using a second single carrier modulated signal and being capable of sending the second signal over any one of a second plurality of frequency bands, the method further comprising the steps of:

a. sending a second line probing signal from the local modem to the remote device over the transmitter channel;

b. receiving the second line probing signal in the remote device;

c. measuring characteristics of the transmitter channel based upon the received second line probing signal; and

d. selecting one of the second plurality of frequency bands based upon the measured characteristics of the transmission channel, said selected one of the second plurality of frequency bands to be used for sending the second modulated signal to the remote device.

70. In a system in which a local modem receives data sent by a remote device over a receiver channel in the form of a first single carrier modulated signal, the modem being capable of receiving the first modulated signal at any one of a first plurality of bit rates, a method for establishing communication conditions comprising the steps of:

a. sending a first line probing signal from the remote device to the local modem over the receiver channel;

b. receiving the first line probing signal in the local modem;

c. measuring characteristics of the receiver channel based upon the received first line probing signal; and

d. selecting one of the first plurality of bit rates based upon the measured characteristics of the receiver channel, said selected one of the first plurality of bit rates to be used for receiving the first modulated signal from the remote device.

71. The method of claim **70** wherein the modem transmits data to the remote device over a transmitter channel by using a second single carrier modulated signal and is capable of sending the second signal at any one of a second plurality of bit rates, the method further comprising the steps of:

a. sending a second line probing signal from the local modem to the remote device over the transmitter channel;

b. receiving the second line probing signal in the remote device;

c. measuring characteristics of the transmitter channel based upon the received second line probing signal; and

d. selecting one of the second plurality of bit rates based upon the measured characteristics of the transmission channel, said selected one of the second plurality of bit rates to be used for sending the second modulated signal to the remote device.

* * * * *

UNITED STATES PATENT AND TRADEMARK OFFICE

CERTIFICATE OF CORRECTION

PATENT NO. : 5,048,054

DATED : September 10, 1991

INVENTOR(S) : Vedat M. Eyuboglu, et al

It is certified that error appears in the above-identified patent and that said Letters Patent is hereby corrected as shown below:

Col. 4, line 14, after "probing", insert --processors--.

Col. 5, line 60, "f_{66}" should be --f_{Δ}--.

Col. 6, example (9), "f_{66}" should be --f_{Δ}--.

Col. 6, example (10), "f_{66}" should be --f_{Δ}--.

Col. 7, example (11), "f_{66}" should be --f_{Δ}--.

Col. 7, line 9, "Ψ"(kf_{Δ})" should be --$\Phi(kf_{\Delta})$--.

Col. 8, line 57, "h'_{nb}" should be --h'_{n}--.

Col. 9, line 7, delete "L" after "between".

Col. 10, line 33, "describe" should be --described--.

UNITED STATES PATENT AND TRADEMARK OFFICE

CERTIFICATE OF CORRECTION

PATENT NO. : 5,048,054

DATED : September 10, 1991

INVENTOR(S) : Vedat M. Eyuboglu, et al

It is certified that error appears in the above-identified patent and that said Letters Patent is hereby corrected as shown below:

Col. 12, line 29, "Q" should be --Q_1--.

Col. 13, line 41, after "phase", insert --.--.

Col. 14, line 2, after "frequencies", insert --,--.

Signed and Sealed this

Fifth Day of January, 1993

Attest:

DOUGLAS B. COMER

Attesting Officer

Acting Commissioner of Patents and Trademarks

EXHIBIT B

U 7042968

THE UNITED STATES OF AMERICA

TO ALL TO WHOM THESE PRESENTS SHALL COME:

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office

December 08, 2006

THIS IS TO CERTIFY THAT ANNEXED HERETO IS A TRUE COPY FROM THE RECORDS OF THIS OFFICE OF:

U.S. PATENT: *5,428,641*
ISSUE DATE: *June 27, 1995*

By Authority of the
Under Secretary of Commerce for Intellectual Property
and Director of the United States Patent and Trademark Office

M. K. CARTER
Certifying Officer

GE 000019

US005428641A

United States Patent [19]

Long

[11] Patent Number: 5,428,641

[45] Date of Patent: Jun. 27, 1995

[54] **DEVICE AND METHOD FOR UTILIZING ZERO-PADDING CONSTELLATION SWITCHING WITH FRAME MAPPING**

[75] Inventor: **Guozhu Long**, Canton, Mass.

[73] Assignee: **Motorola, Inc.**, Schaumburg, Ill.

[21] Appl. No.: **97,343**

[22] Filed: **Jul. 23, 1993**

[51] **Int. Cl.**[6] **H04L 27/04**

[52] **U.S. Cl.** **375/295**; 371/43

[58] **Field of Search** 375/17, 69, 39, 42, 375/94; 371/37.8, 43

[56] **References Cited**

U.S. PATENT DOCUMENTS

4,597,090	6/1986	Forney, Jr.	375/39
4,807,253	2/1989	Hagenauer et al.	371/43
5,029,185	7/1991	Wei	375/27
5,233,629	8/1993	Pair et al.	375/39

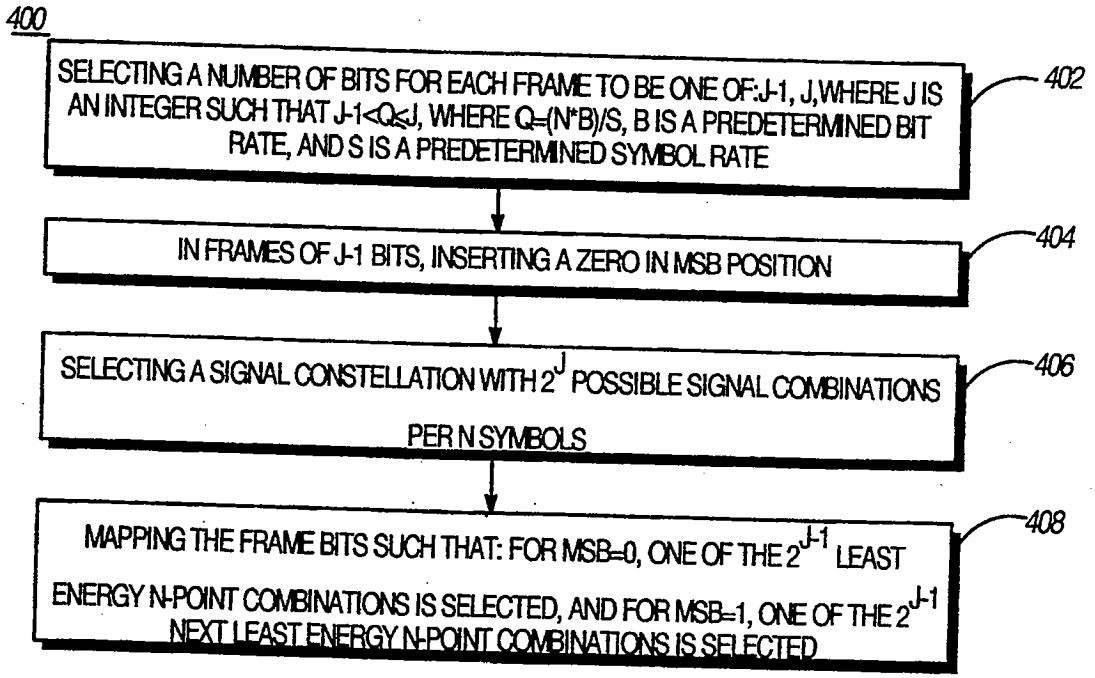
Primary Examiner—Young Tse

Attorney, Agent, or Firm—Darleen J. Stockley

[57] **ABSTRACT**

A device (500) and method (400) for zero-padding constellation switching with frame mapping provides reduced complexity for mapping frames having possibly a fractional number of bits and a predetermined number of symbols while eliminating the usual disadvantages of constellation switching.

8 Claims, 2 Drawing Sheets



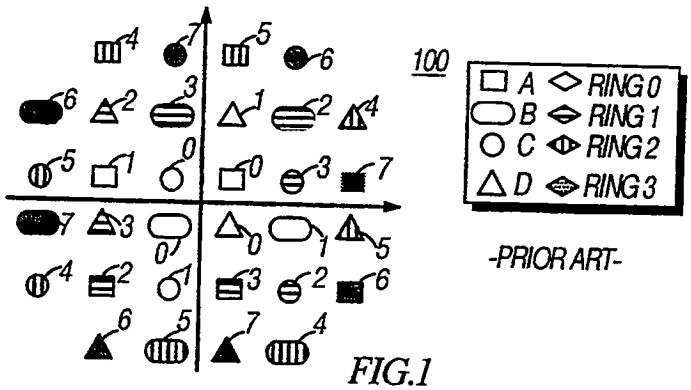


FIG.1

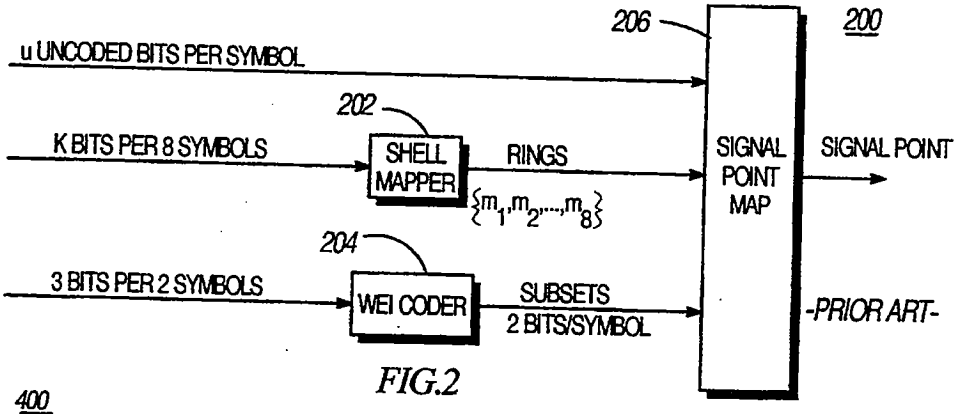


FIG.2

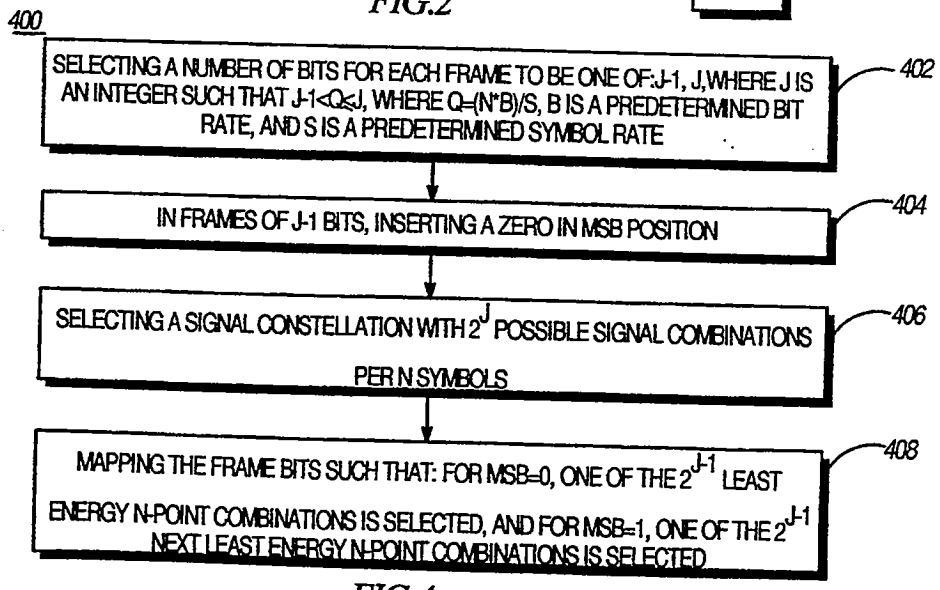


FIG.4

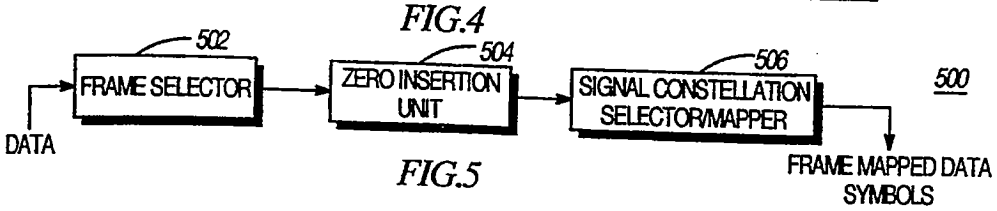
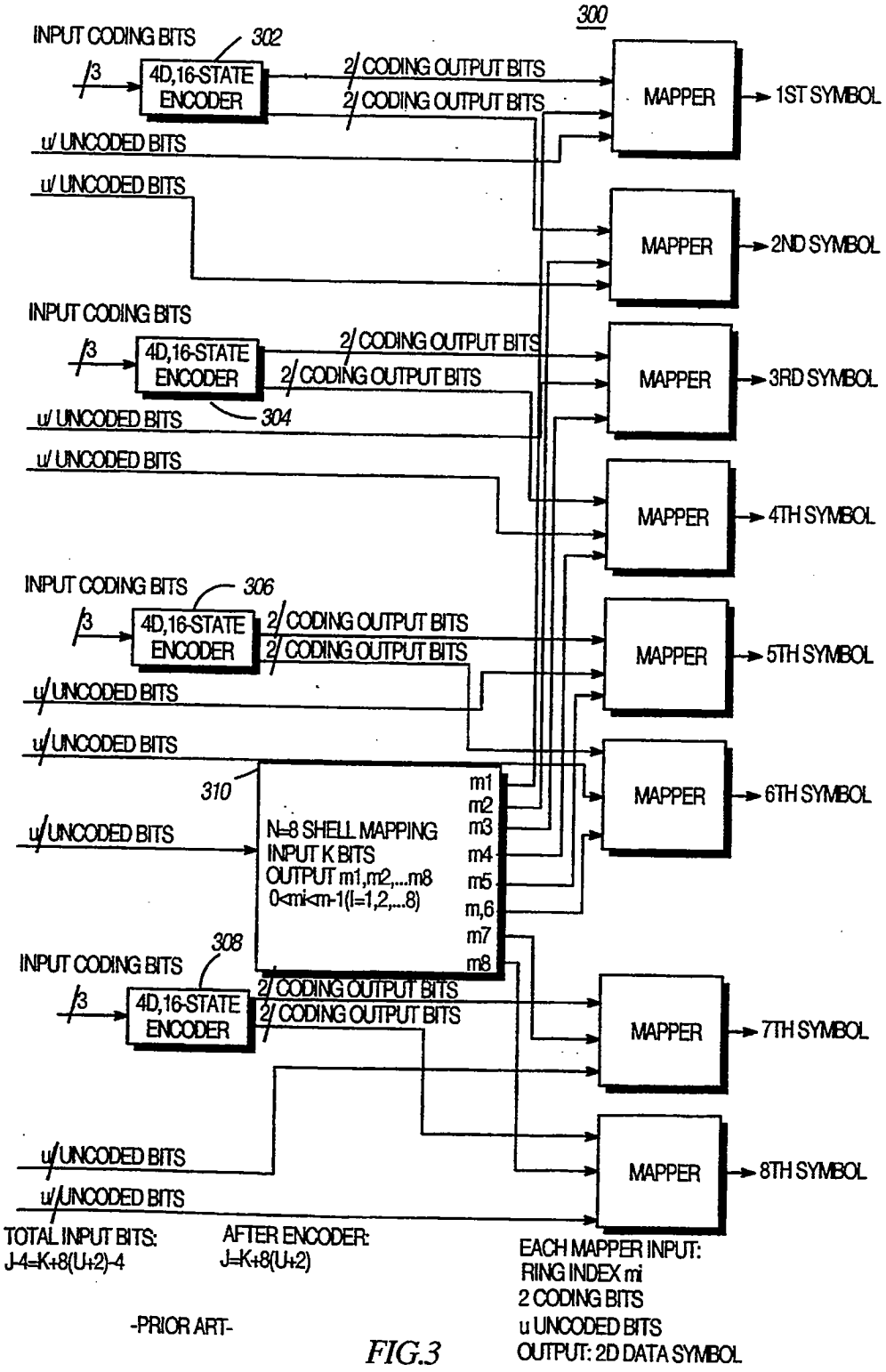


FIG.5



-PRIOR ART-

FIG.3

DEVICE AND METHOD FOR UTILIZING ZERO-PADDING CONSTELLATION SWITCHING WITH FRAME MAPPING

FIELD OF THE INVENTION

The present invention relates generally to digital communication devices and methods, and more particularly to mapping a digital data sequence for transmission in a digital communication system.

BACKGROUND

Modems are typically utilized for transmitting and receiving digital data communication from computer systems. Typically, data is expressed in a binary form. Data is transmitted at a bit rate, e.g., B bits per second, where the bit rate is defined as the number of bits to be transmitted and received, including the actual binary information data rate and a predetermined redundancy needed by coding in a selected system. During transmission, the binary data information is typically transmitted and received in a form of a series of symbols at a symbol rate of S symbols per second. Thus, each symbol contains B/S bits of binary data.

Each symbol can be represented by one of the possible line signal states generated by the modem. Various modulation techniques may be used to convert data into line signal states. For example, in quadrature amplitude modulation (QAM), The line signal states can be represented by a set of complex numbers, namely by a set of points in a two-dimensional signal constellation. For example, for a bit rate B and a symbol rate S, where B/S is an integer D, a signal constellation of size 2^D is needed to represent D bits in each symbol. Thus, if B=12000 bits/second and the symbol rate S is 2400 symbols/second, D=5 bits/symbol, and a 32-point two-dimensional signal constellation is used, providing a scheme for mapping one out of 32 possible complex signal points according to 5 input data bits.

However, for high speed modems, multiple symbol rates and bit rates may be used to facilitate more efficient use of the available channel bandwidth. In such instances, the ratio B/S may not be always an integer.

Where B/S is not an integer, the modem transmits a fractional number of bits per symbol. Various techniques have been used to accomplish transmission of a fractional number of bits per symbol. In the constellation switching technique, for example, where $D-1<B/S<D$, the modem switches between D-1 bits/symbol and D bits/symbol such that, on an average, the modem sends B/S bits per symbol. Namely, the modem switches between a signal constellation with 2^{D-1} points and another signal constellation with 2^D points.

The main disadvantages of constellation switching are the introduction of a variation in constellation size and an increase in peak-to-average power ratio due to the increase in constellation size. This is undesirable in many applications, especially when the communication channel introduces signal-dependent impairments such as harmonic distortion and pulse coded modulation (PCM) noise. In addition, constellation switching often complicates the modem implementation.

In contrast to one symbol by one symbol mapping techniques, frame-mapping techniques may be used.

Since B/S is usually a rational number, there exists a number N such that Q=N*B/S is an integer, i.e., the number of bits in a frame of N symbols is an integer. A frame-mapping algorithm maps the incoming Q bits to N symbols chosen from a signal constellation with a sufficiently large number of points. Compared with the constellation required by symbol-based constelation switching (with 2^D points), frame-mapping techniques usually require a relatively small constellation, and thus a small peak-to-average power ratio. However, such techniques generally introduce complexity where a large N is used. Examples of frame-mapping techniques include modulus conversion and shell mapping.

Since multiple B and S values are often selected, there is often difficulty in selecting a reasonably small N such that Q=N*B/S is always an integer for all possible combinations of B and S. In some instances, a different N may be selected for different combinations of B and S, thereby further complicating the implementation of the mapping algorithm. Another approach is to transmit a fractional number of bits per frame, i.e., using a fixed N in conjunction with a constellation switching among frames. Both approaches provide more implementation complexity.

Hence, there is a need for a frame-mapping device and method which maps data that is transmitted at fractional bits per frame rate such that the implementation difficulties of constellation switching are avoided.

BRIEF DESCRIPTIONS OF THE DRAWINGS

FIG. 1 is an exemplary signal constellation, as is known in the art.

FIG. 2 is a block diagram of a shell mapper as is known in the art.

FIG. 3 is a block diagram of a 4 dimensional/16-state encoder that is configured for shell mapping with N=8, as is known in the art.

FIG. 4 is a flow chart showing the steps of an embodiment of the frame-mapping method of the present invention.

FIG. 5 is a block diagram of a frame-mapping device in accordance with an embodiment of the present invention.

DETAILED DESCRIPTION OF A PREFERRED EMBODIMENT

The present invention provides an advantage for mapping frames of data which would otherwise require constellation switching by utilizing a scheme of selecting first and second frame sizes for data, zero-padding the data for the first frame size such that the first frame is equal to the second frame size, and frame mapping the data.

This scheme is particularly advantageous in that it eliminates the necessity for constellation switching when Q, an average number of bits per frame of N symbols where Q=N,B/S (N is a predetermined integer, B is a number of bits per frame, S is a symbol rate) is expressible in a form of an improper fraction, hereafter referred to as fractional.

Where a frame-mapping technique is used to map a frame of Q=N*B/S bits into N symbols, if Q, the number of bits in a frame of N symbols, is an integer, a frame-mapping algorithm maps the incoming Q bits to N symbols chosen from a signal constellation with a sufficiently large number of points. Examples of frame-mapping techniques include modulus conversion and shell mapping.

In shell mapping, the selected signal constellation is divided into M equal size "rings" each of which contains R signal points. R is typically chosen to be 2^r,

where v is an integer greater than or equal to zero. Among Q bits, K=Q−(N,v) bits, are used for shell mapping, generating N ring indices ranging from 0 to M-1. In each symbol, one of these ring indices is used to choose one of the M rings in the signal constellation, and v bits are used to choose one of the 2^v points within that ring. If a convolutional encoder is employed, among v bits, c bits are coding bits which are the output of the convolutional encoder, and are used to select one of the 2^c subsets, and the remaining u=v−c bits are used to select one of the 2^u points in the chosen subset in the chosen ring. When no convolutional encoder is employed, c=0 and u=v.

For example, for a 4-dimensional 16-state code, c=2, and the constellation is partitioned into 2^c=4 equal-sized subsets A,B,C and D. The constellation is also partitioned into M equal-sized rings, each of which contains 2^{u+c} points, of which 2^u points are from each subset. Ring **0** contains the 2^u lowest energy points from each subset, ring **1** contains the 2^u next lowest energy points from each subset, and so on. An exemplary 32 point signal constellation is shown in FIG. **1**, numeral 100, where L=32, c=2, u=1 and M=4. These 32 points may be divided into 4 rings (shown as rings 0 (unshaded), 1 (shaded with horizontal lines), 2 (shaded with vertical lines), 3 (dotted)), each of which consists of 8 points (indexed 0–1, or 2–3, or 4–5, or 6–7) of which 2 points are from each subset. The energy of the signal point is non-decreasing as the index increases. The collection of all 32 points with index 0 to 7 is approximately within a circle.

The constellation size is $L=M*2^{u+c}$.

Shell mapping uses K=Q−(N*v) bits to generate N ring indices m_i (i=1,2, . . .N) ranging from 0 to M-1. In order to select the N-tuple $\{m_1, m_2, \ldots m_N\}$ using the K bits, first a "cost" is assigned to each ring. Since a main purpose of an efficient mapping scheme is to achieve a low average signal power, the average signal power within a ring is assigned as its "cost". However, for computational convenience, the "cost" is set to the ring index. Thus, the innermost ring has cost **0**, the next ring has cost **1**, and so on. This approximation is quite workable, in particular as the constellation becomes larger. Since costs are additive, the total costs of N symbols is $m_1+m_2 \ldots +m_N$. Note that there are M^N combinations of N ring indices, but only $2^K \leqq M^N$ input K bit combinations. Thus, certain combinations of rings will be excluded. An efficient shell mapping scheme minimizes the cost by selecting the 2^K combinations that have the least total cost. If the input K bits is expressed as a number X, an efficient shell mapping algorithm ensures that as X increases, the corresponding total cost is non-decreasing. As a result, when the most significant bit in the incoming K bits is 0, one of the 2^{K-1} combinations with least total costs will be selected; and when the most significant bit in the incoming K bits is 1, one of the 2^{K-1} combinations with next least total costs will be selected. It is this property that makes this invention possible.

The integer M is chosen such that $2^K \leqq M^N$. The smallest integer for M that satisfies this condition is defined as M_{min}.

When $M=M_{min}$, the constellation size is $L=M_{min}*2^{u+c}$, which is in general smaller than the constellation size required by symbol-based constellation switching, thus the peak power is reduced. When M is selected such that $M>M_{min}$, the difference between 2^K and M^N is larger. Interestingly, in this case, the 2^K least energy ring index combinations will have a smaller average total cost. Thus, using the shell mapping algorithm with a larger than the minimum constellation allows reduction of the average cost (shaping gain) at the expense of constellation expansion.

The shell mapping scheme establishes a 1-1 mapping between K binary bits and an N-tuple $\{m_1, m_2, \ldots m_N\}$. Though this mapping may be done using a table look-up, such a mapping table is often too large to be practical. Instead, some efficient mapping algorithms can be used to achieve the mapping such as the shell mapping technique set forth below.

Since multiple B and S values are often selected, there is often difficulty in selecting a reasonably small N such that Q=N*B/S is always an integer for all possible combinations of B and S. One approach is to fix N for all the cases, but this requires transmitting a fractional number of bits per frame in some of the cases.

Where Q is fractional, an integer value J can always be selected such that J-1<Q≦J. J-1 bits are transmitted in some of the frames and J bits are transmitted in the rest of the frames to achieve an average rate of Q bits per frame. Conventional, this requires a constellation switching among frames. Using the scheme of the present invention, such a constellation switching can be avoided.

The values for J-1 and J represent first and second frame sizes, i.e., number of bits per frame, for incoming data. The key to the invention is that only one signal constellation is selected that is suitable for mapping J bits/frame. That is, a signal constellation has at least 2^J possible signal point combinations per N symbols, i.e., providing transmission of J/N bits per symbol. The frame mapping of the data to the signal points provides that where, in the transmitted data frame, the most significant bit (MSB) of the J bits is equal to zero, one of the 2^{J-1} least energy combinations of N signal points is selected. In frames having only J-1 bits, a zero is inserted at the MSB position such that the J-1 bits frame then contains J bits. The frame mapping scheme is the same for all frames of data, ensuring a less complex implementation than if constellation switching were utilized to achieve mapping for a fractional bit/frame rate. Typical values for N are integers 2 and greater. This scheme is particularly useful for frame sizes of 4 or greater.

Further, where the frame mapping is shell mapping, the present invention is very efficient in providing a minimized constellation expansion and a small average signal power.

To further explain shell mapping, the following exemplary shell mapping technique describes an implementation of shell mapping for a trellis code that is a 4D 16-state Wei code where the frame size is N=8.

An L-point signal constellation B is partitioned into four subsets B_0, B_1, B_2, B_3 according to Ungerboeck set partitioning principles, as is known in the art. B is also partitioned into M equal-sized "rings", where ring 0 contains the 2^u least-energy points from each subset, ring **1** contains the 2^u next lowest energy points, and so forth up to ring M−1. It follows, as above, that $L=M*2^{u+2}$.

In this example, where there are I input bits per frame of 8 symbols, 12 input bits of the I bits are used as the 4D/16-state encoder input, and 8u bits, u an integer, are used as uncoded bits. After encoding, 4 redundant bits are added such that there are a total of J=I+4 bits. As shown in FIG. **2**, numeral **200**, a block diagram of a

shell mapper as is known in the art, from the J bits, K=J-8(u+2) bits are used as the input to a shell mapper (202) to select the ring labels $\{m_1, m_2, \ldots, m_8\}$. Two bits per symbol from the Wei coder (204), which includes differential encoding, select a subset B_i, and the remaining u bits per symbol select the signal point in the selected ring of a signal point map (206). To limit complexity, the integer u is selected such that K is as large as possible, but no larger than a predetermined limit K_{max}.

Selection of the ring labels $\{m_1, m_2, \ldots, m_8\}$ is very important in shell mapping. First, a cost is assigned to each ring, typically the cost c(i) of the i'th ring being equal to its label i. The total cost of a combination of rings is the sum of the costs of the individual rings. There are M^8 ring combinations, but only $2K \leqq M^8$ input bit combinations. The shell mapper (202) selects the 2K combinations that have the least total cost to minimize the average cost.

For example, where $G_1(z)$is a generating function of costs of a single ring; i.e.,

$$G_1(z)=\Sigma_i a_1(i)z^1=1+z+z^2+\ldots+z^{M-1}.$$

The coefficient $a_1(i)$ of z^i represents the number of rings that have cost i; namely, $a_1(i)=1$ if $0 \leqq i \leqq M-1$, and $a_1(i)=0$ otherwise. Similarly, where $G_{2j}(z)$ be the generating function of costs of 2^j-ring combinations, where j=1, 2 or 3; i.e.,

$$G_{2j}(z)=\Sigma_i a_{2j}(i)z^i,$$

it is seen that

$$G_8(z)=[G_4(z)]^2=[G_2(z)]^4=[G_1(z)]^8.$$

$z_8(i)$ is defined as the number of eight-ring combinations with cost less than i:

$$z_8(i)=\Sigma 0 \leqq j \leqq a_8(j).$$

Where $a_2(i)$, $a_4(i)$and $z_8(i)$ are precomputed and stored in memory, the following steps are utilized:

(1) Collect the K input bits and represent them as an integer X, $0 \leqq X \leqq 2^K-1$.

(2) Find the largest integer C_8, $0 \leqq C_8 \leqq 8(M-1)$, for which $z_8(C_8) \leqq X$. This can be done by a binary search through the table for $z_8(i)$. The value C_8 is the total cost of $(m_1, m_2, \ldots, m_8)$.

(3) Compute the residue $R_8=X-z_8(C_8)$, where $0 \leqq R_8 \leqq g_8(C_8)-1$. Then use C_8 and R_8 to determine $(m_1, m_2, \ldots, m_8)$ as follows:

First, split C_8 into C_{41} and C_{42}, which are the total costs of (m_1, m_2, m_3, m_4) and (m_5, m_6, m_7, m_8), respectively. Since $C_{41}+C_{42}=C_8$, the following combinations are obtained:

C_{41}	C_{42}	Number of Combinations
0	C8	$a_4(0)a_4(C_8)$
1	C_8-1	$a_4(1)a_4(C_8-1)$
2	C_8-2	$a_4(2)a_4(C_8-2)$
...		
C_8	0	$a_4(C_8)a_4(0)$

It follows that if the convolution terms $a_4(0)a_4(C_8)$, $a_4(1)a_4(C_8-1), \ldots$, are subtracted from R_8 in sequence according to

$$R_8=R_8-a_4(s)a_4(C_8-s),\ s=0, 1, \ldots, C_8,$$

then within at most C_8+1 iterations R_8 will become negative (similar to FIR filtering). When $R_8<0$, the value of s is recorded as C_{41} and C_8-s is denoted as C_{42}. C_{41} and C_{42} represent the costs of the four-tuples (m_1, m_2, m_3, m_4) and (m_5, m_6, m_7, m_8), respectively. Then the last term $a_4(C_{41})a_4(C_{42})$ is added to R_8 to obtain the new residue R_4, $0 \leqq R_4 \leqq a_4(C_{41})a_4(C_{42})-1$, and R_4 is represented as:

$$R_4=R_{42}a_4(C_{41})+R_{41}$$

where $0 \leqq R_{41} \leqq a_4(C_{41})-1$ and $0 \leqq R_{42} \leqq a_4(C_{42})-1$. R_{41} and R_{42} are obtained by dividing R_4 by $a_4(C_{41})$. The quotient is R_{42} and the remainder is $R_{41}=R_4-R_{42}$ $a_4(C_{41})$.

To obtain the four-tuple (m_1, m_2, m_3, m_4) from the residue R_{41} and the cost C_{41}. C_{41} is split into two components, C_{211} and C_{212}, which are the costs of the two-tuples (m_1, m_2) and (m_3, m_4), respectively.

Since $G_4(z)$ is the convolution of $G_2(z)$ with itself, the. procedure above can be used; i.e., subtract from R_{41} the convolution terms $a_2(0)a_2(C_{41})$, $a_2(1)a_2(C_{41}-1)$, . . . in sequence according to

$$R_{41}=R_{41}-a_2(j)a_2(C_{41}-j),\ j=0, 1, \ldots, C_{41},$$

until R_{41} becomes negative (within at most $C_{41}+1$ iterations). Then the values of j and $C_{41}-j$ are recorded as C_{211} and C_{212} the costs of the pairs (m_1, m_2) and (m_3, m_4), respectively. The last term $a_2(C_{211})a_2(C_{212})$ is added back to R_{41} to obtain the new residue R_{21}, $0 \leqq R_{21} \leqq a_2(C_{211})a_2(C_{212})-1$. R_{21} is then represented in the form

$$R_{21}=R_{212}a_2(C_{211})+R_{211}$$

where $0<R_{211} \leqq a_2(C_{211})-1$ and $0 \leqq R_{212} \leqq a_2(C_{212})-1$. The quantities R_{211} and R_{212} are obtained, as before, by dividing R_{21} by $a_2(C_{211})$.

Determining the rings $(m_1, m_2, m_3, m_4)$is now straightforward; since $a_1(i)=1$, $0 \leqq i \leqq M-1$, it is clear that:

(a) If $C_{211} \leqq M-1$, then $m_1=R_{211}$ and $m_2=C_{211}-R_{211}$.

(b) If $M-1 \leqq C_{211} \leqq 2(M-1)$, then $m_1=C_{211}-(M-1)+R_{211}$ and $m_2=M-1-R_{211}$.

(c) If $C_{212} \leqq M-1$, then $m_3=R_{212}$ and $m_4=C_{212}-R_{212}$.

(d) If $M-1<C_{212} \leqq 2(M-1)$, then, $m_3=C_{212}-(M-1)+R_{212}$ and $m_4-M-1-R_{212}$.

Similarly, the four-tuple (m_5, m_6, m_7, m_8) may be obtained from R_{42} and C_{42}.

Thus, the original K bits are uniquely recovered in a receiver from the received eight-tuple $(m_1, m_2, \ldots, m_8)$. The operation of the decoder is easily derived from that of the encoder.

FIG. 3, numeral 300, is a block diagram of a 4-dimensional/16-state encoder that is configured for shell mapping with N=8, as is known in the art. In this embodiment of shell mapping, which is suitable for the shell mapping portion of the present invention, two bits per symbol from the convolutional encoder (302, 304, 306, 308) select the subset A, B C or D. Then the shell mapper (310) utilizes K bits per frame of eight symbols to generate 8 ring indices for selecting rings, and the re-

maining u bits per symbol select the signal point in the appropriate subset within the selected ring.

FIG. 4, numeral 400, is a flow chart showing the steps of an embodiment of the frame-mapping method of the present invention. The frame-mapping method is utilized for mapping N-symbol frames of data, N a predetermined integer.

The first step (402) is selecting a number of bits for each frame to be one of: J-1, J, where J is an integer such that $J-1<Q\leqq J$, where $Q=N*B/S$, B is a predetermined bit rate, and S is a predetermined symbol rate. Then, in frames of J-1 bits, a zero is inserted (404) in a most significant bit (MSB) position. A signal constellation is selected (406) with at least 2^J possible signal combinations per N symbols. Then, the frame bits are mapped (408) such that for MSB-0, one of the 2^{J-1} least energy combinations of N points is selected from the signal constellation such that the averaged energy is minimized. Note that when Q is an integer, Q=J, and all the frames have J bits. Thus, the mapping reduces to conventional frame mapping for integer number of bits per frame.

When using shell mapping, among J bits, K=J-N,v bits are used for shell mapping which generates N ring indices ranging from 0 to M-1. In each symbol, one of these ring indices is used to choose one of the M rings in the signal constellation, and v bits ($v\leqq 0$, an integer) are used to choose one of the 2^v points within that ring. Among v bits, c bits ($c\leqq 0$, an integer) are coding bits which are the output of the convolutional encoder, and are used to select one of the 2^c subsets, and the remaining u=v-c bits($u\leqq 0$, an integer) are used to select one of the 2^u points in the chosen subset in the chosen ring.

Choosing a large K will normally increase the complexity of the shell mapping scheme. Usually, K is limited to less than or equal to K_{max}. The typical values for K_{max} are, for example, 15 or 31. In cases where J is large, K is kept to be less than or equal to K_{max} by selecting a large u. Thus, if B is a two-dimensional signal constellation with $L=M*2^{u+c}$ points, where M is an integer such that $2^{K/N}\leqq M$. M_{min} is defined to be the minimum value for the integer M. M may be selected such that $M>M_{min}$ to obtain more shaping gain and such that $M=M_{min}$ to obtain the smallest constellation size. Selecting a different value for M allows a trade-off between shaping gain and constellation size.

In the embodiment of the method of the present invention, Q is a fractional bits per symbol rate. When the frame-mapping method is shell mapping, the signal constellation is divided into M equal size rings each of which has 2^v ($v\leqq 0$, a predetermined integer) points. In this embodiment, the number of bits in each frame is one of: J-1 and J.

Where J-1 is the number of bits in the frame, K-1 ($K=J-N*v$, a predetermined integer) bits, together with a zero as the MSB, are utilized for shell mapping, to obtain N ring indices ranging from 0 to M-1 (M an integer) such that an averaged sum of N ring indices obtained in shell mapping is minimized, thereby minimizing average signal power. Where J is the number of bits in the frame, K bits are utilized for shell mapping to obtain N ring indices.

FIG. 5, numeral 500, is a block diagram of a frame-mapping device in accordance with the embodiment of the present invention. The frame-mapping device maps N-symbol frames of data, N a predetermined integer, such that a fractional number of bits per frame can be transmitted without constellation switching, and includes a frame selector (502), a zero insertion unit (504), and a signal constellation selector/mapper (506). The frame selector (502) is operably coupled to receive the data, for selecting a number of bits for each frame to be one of: J-1, J, where J is an integer such that $J-1\leqq Q<J$, where $Q=N*B/S$, B is a predetermined bit rate, and S is a predetermined symbol rate. The zero insertion unit (504) is operably coupled to the frame selector (502), for, in frames of J-1 bits, inserting a zero in a most significant bit (MSB) position. The signal constellation selector/mapper (506) is operably coupled to the zero insertion unit (504), for selecting a signal constellation with at least 2^J possible signal combinations per N symbols, and mapping the frame bits such that for MSB=0, one of the 2^{J-1} least energy N-point combinations is chosen from the signal constellation so that the average energy is minimized. The frame-mapping device operates in accordance with the frame-mapping method described above.

Although exemplary embodiments are described above, it will be obvious to those skilled in the art that many alterations and modifications may be made without departing from the invention. Accordingly, it is intended that all such alterations and modifications be included within the spirit and scope of the invention as defined in the appended claims.

I claim:

1. A frame-mapping method for mapping N-symbol frames of data, N a predetermined integer ($N>1$), such that a fractional number Q of bits per frame can be transmitted without constellation switching, comprising the steps of:

A) selecting a number of bits for each frame to be one of: J-1, J, where J is an integer such that $J-1<Q<J$, where $Q=N*B/S$, B is a predetermined bit rate, and S is a predetermined symbol rate,

B) in frames of J-1 bits, inserting a zero in a most significant bit (MSB) position,

C) selecting a signal constellation with 2^J possible signal combinations per N symbols, and

D) mapping the frame bits such that for MSB=0, one of the 2^{J-1} N-point combinations with least average energy is selected from the signal constellation.

2. The frame-mapping method of claim 1 wherein:

A) the frame-mapping method is shell mapping,

B) the signal constellation is divided into M equal size rings, M an integer, each of which has 2^v ($v\leqq 0$ a predetermined integer) points, and

C) the number of bits in each frame is one of: J-1 and J,

D) where J-1 is the number of bits in the frame, K-1 ($K=J-N*v$) bits, together with a zero as the MSB, are utilized for shell mapping, to obtain N ring indices ranging from 0 to M-1 (M being an integer) such that an average sum of N ring indices obtained in shell mapping is minimized, thereby minimizing average signal power, and

E) where J is the total number of bits in the frame, K bits are utilized for shell mapping to obtain N ring indices.

3. A frame-mapping device for mapping N-symbol frames of data, N a predetermined integer ($N>1$), such that a fractional number of bits per frame can be transmitted without constellation switching, comprising:

A) a frame selector, operably coupled to receive the data, for selecting a number of bits for each frame of data to be one of: J-1, J, where J is an integer such that $J-1<Q\leqq J$, where $Q=N*B/S$, B is a

predetermined bit rate, and S is a predetermined symbol rate,

B) a zero insertion unit, operably coupled to the frame selector, for, in frames of J-1 bits, inserting a zero in a most significant bit (MSB) position,

C) a signal constellation selector/mapper, operably coupled to the zero insertion unit, for selecting a signal constellation with at least 2^J possible signal combinations per N symbols, and mapping the frame bits such that for MSB=0, one of 2^{J-1} combinations of N points with least average energy is selected from the signal constellation.

4. The frame-mapping device of claim 3 wherein:

A) the frame-mapping device is a shell mapper,

B) the signal constellation is divided into M equal size rings. M an integer, each of which has 2^v ($v \geqq 0$, a predetermined integer) points, and

C) the number of bits in each frame is one of: J-1 and J,

D) where J-1 is the number of bits in the frame, K-1 (K=J-N*y, an integer) bits, together with a zero as the MSB, are utilized for shell mapping, to obtain N ring indices, the indices ranging from 0 to M-1 (M being an integer) such that an average sum of N ring indices obtained in shell mapping is minimized, thereby minimizing average signal power, and

E) where J is the total number of bits in the frame, K bits are utilized for shell mapping to obtain N ring indices.

5. A frame-mapping method for mapping successive frames of data to groups of N symbols, N a predetermined integer (N>1), such that, on average, a fractional number Q of bits are mappable per frame without constellation switching, comprising the steps of:

A) selecting a number of bits for each frame to be one of: J-1, J, where J is an integer such that J-1<Q<J, according to a predetermined pattern,

B) in frames of J-1 bits, inserting a zero in a most significant bit (MSB) position,

C) selecting a set of 2^J possible combinations of N symbols, where each symbol is chosen from a signal constellation and

D) mapping the frame bits such that for MSB=0, one of the 2^{J-1} possible combinations of N symbols of least average energy is selected from the 2^J possible combinations.

6. The frame-mapping method of claim 5 wherein:

A) the frame-mapping method is shell mapping,

B) the signal constellation is divided into M equal size rings, M an integer, each of which has 2^v ($v>0$ a predetermined integer) points, and

C) where in frames of J-1 bits, K-1 (K=J-N*v) bits, together with a zero as the MSB, are utilized for shell mapping, to obtain N ring indices ranging from 0 to M-1 such that an average sum of N ring indices obtained in shell mapping is kept small, thereby keeping the average signal power small, and

D) where in frames of J bits, K bits are utilized for shell mapping to obtain N ring indices.

7. A frame-mapping device for mapping successive frames of data to groups of N symbols, N a predetermined integer (N>1), such that, on average, a fractional number Q of bits are mappable per frame without constellation switching, comprising:

A) a frame selector, operably coupled to receive the data, for selecting a number of bits for each frame of data to be one of: J-1, J, where J is an integer such that J-1<Q<J, according to a predetermined pattern,

B) a zero insertion unit, operably coupled to the frame selector, for, in frames of J-1 bits, inserting a zero in a most significant bit (MSB) position,

C) a signal constellation selector/mapper, operably coupled to the zero insertion unit, for selecting a set of 2^J possible combinations of N symbols, where each symbol is chosen from a signal constellation, and mapping the frame bits such that for MSB=0, one of 2^{J-1} possible combinations of N symbols of least average energy is selected from the set of 2^J possible combinations.

8. The frame-mapping device of claim 7 wherein:

A) the frame-mapping device is a shell mapper,

B) the number of bits in each frame is one of: J-1 and J,

C) where in frames of J-1 bits, K-1 (K=J*N-v) bits, together with a zero as the MSB, are utilized for shell mapping to obtain N ring indices ranging from 0 to M-1 (M being an integer) such that an average sum of N ring indices obtained in shell mapping is minimized, thereby minimizing an average signal power, and

D) where in frames having a total of J bits, K bits are utilized for shell mapping to obtain N ring indices.

* * * * *

UNITED STATES PATENT AND TRADEMARK OFFICE

CERTIFICATE OF CORRECTION

PATENT NO. : 5,428,641 Page 1 of 2

DATED : Jun. 27, 1995

INVENTOR(S) : Guozhu Long, Canton, Mass.

It is certified that error appears in the above-indentified patent and that said Letters Patent is hereby corrected as shown below:

Column 5:

line16 reads "2K"	should read "2^K"
line17 reads "2K"	should read "2^K"
line 23 reads "$= \Sigma_i a_1(i)z^1$"	should read "$= \Sigma_i a_1(i)z^i$"

Column 7:

line 23 reads " K= J-N,v "	should read " K = J-N*v "
line 27 reads " (v ≦ 0, an interger)"	should read "(v ≥ 0, an interger) "
line 29 reads " (c ≦ 0, an interger)"	should read "(c ≥ 0, an interger) "
line 32 reads " (u ≦ 0, an interger)"	should read "(u ≥ 0, an interger) "

UNITED STATES PATENT AND TRADEMARK OFFICE

CERTIFICATE OF CORRECTION

PATENT NO. : 5,428,641
DATED : Jun. 27, 1995
INVENTOR(S) : Guozhu Long, Canton, Mass.

It is certified that error appears in the above-indentified patent and that said Letters Patent is hereby corrected as shown below:

Column 8:

line 47 reads " (v $\leqq$ 0, a predetermined" should read "(v $\geq$ 0, a predetermined "

Column 9

line 21 reads "K=J-N*y" should read "K=J-N*v"

Column 10

line 39 reads "K=J*N-v" should read "K=J-N*v"

Signed and Sealed this
Twenty-eighth Day of January, 1997

Attest:

BRUCE LEHMAN

Attesting Officer *Commissioner of Patents and Trademarks*

EXHIBIT C

U 7042968

THE UNITED STATES OF AMERICA

TO ALL TO WHOM THESE PRESENTS SHALL COME:

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office

December 08, 2006

THIS IS TO CERTIFY THAT ANNEXED HERETO IS A TRUE COPY FROM THE RECORDS OF THIS OFFICE OF:

U.S. PATENT: *5,446,758*
ISSUE DATE: *August 29, 1995*

By Authority of the
Under Secretary of Commerce for Intellectual Property
and Director of the United States Patent and Trademark Office

M. K. CARTER
Certifying Officer

GE 000030

US005446758A

United States Patent [19]

Eyuboglu

[11] Patent Number: 5,446,758

[45] Date of Patent: Aug. 29, 1995

[54] **DEVICE AND METHOD FOR PRECODING**

[75] Inventor: **M. Vedat Eyuboglu**, Concord, Mass.

[73] Assignee: **Motorola, Inc.**, Schaumburg, Ill.

[21] Appl. No.: **89,319**

[22] Filed: **Jul. 8, 1993**

[51] **Int. Cl.**[6] **H04B 1/10**; H04L 5/12; H04L 27/00; G06F 11/10

[52] **U.S. Cl.** **375/259**; 375/265; 375/340; 371/43

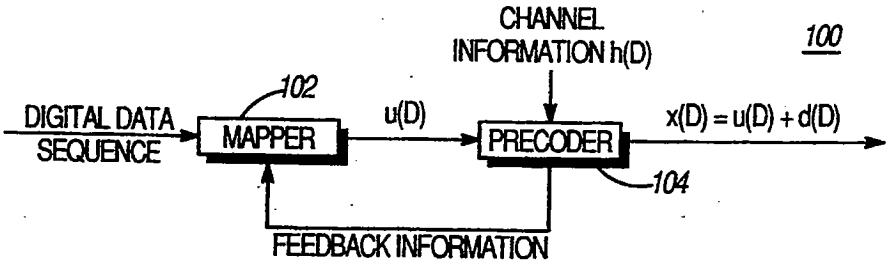
[58] **Field of Search** 375/39, 94, 37; 371/43

Primary Examiner—Edward L. Coles, Sr.
Assistant Examiner—John Ning
Attorney, Agent, or Firm—Darleen J. Stockley

[57] **ABSTRACT**

An improved precoding technique (**700**) and device (**100**) allows transmission of a signal point sequence over a channel h(D) to provide efficient data transfer in the presence of intersymbol interference and noise at data rates approaching channel capacity. This improved technique works with trellis-coded modulation and with any signal constellation. Thus, the present invention simplifies shaping and allows signaling at fractional rates without constellation switching. A key advantage of the present invention over prior art is its ability to achieve reduced dither loss by selecting the output components of a mapper based on past components of a channel output sequence.

42 Claims, 3 Drawing Sheets



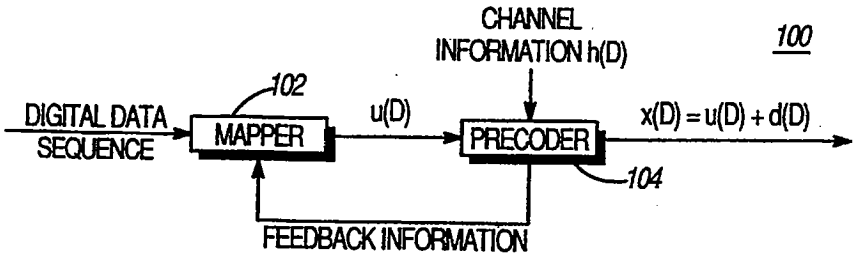


Fig.1

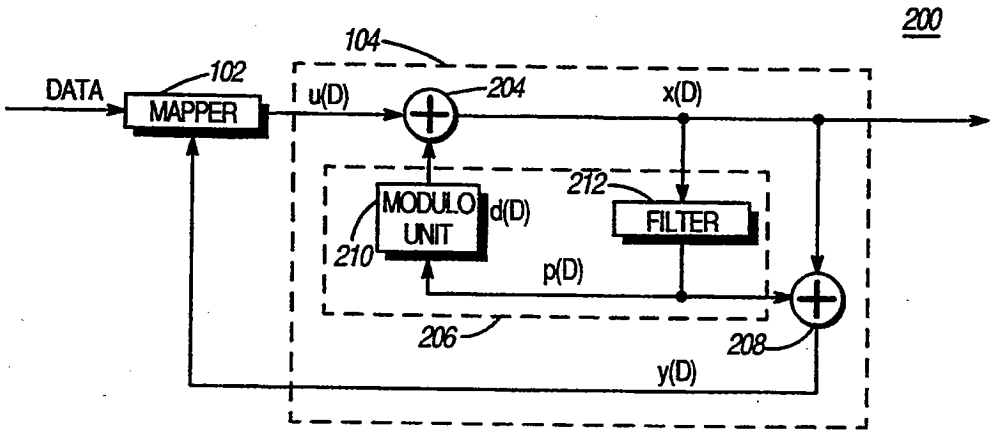


Fig.2

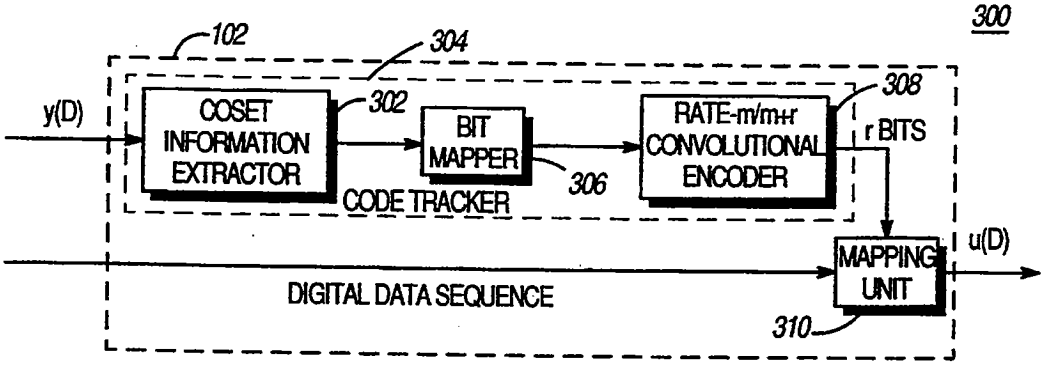


Fig.3

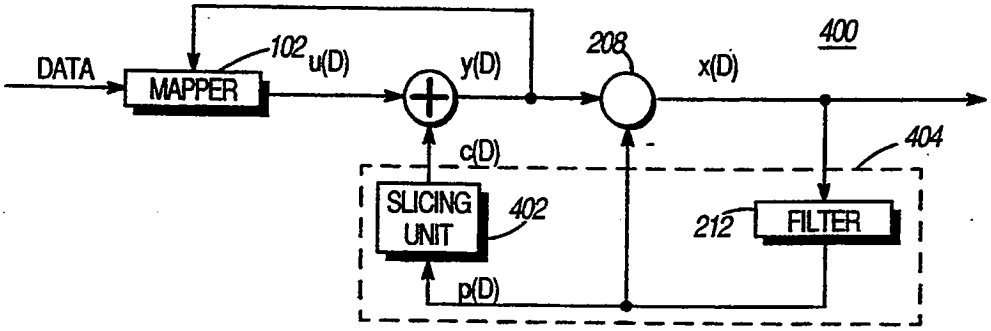


Fig.4

GE 000032

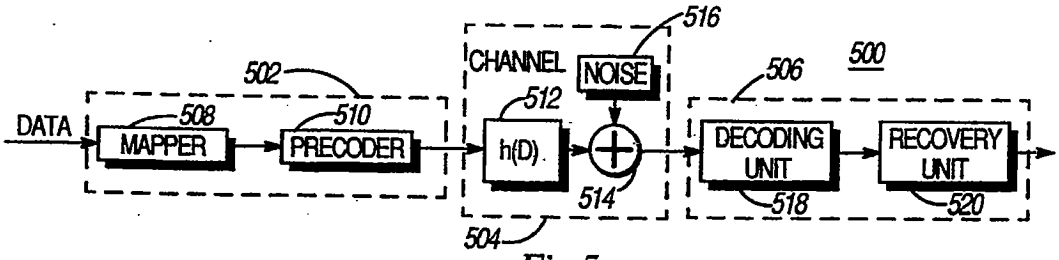


Fig.5

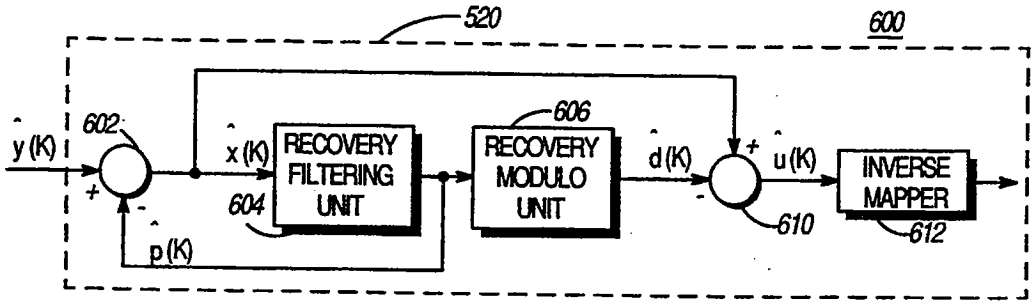


Fig.6

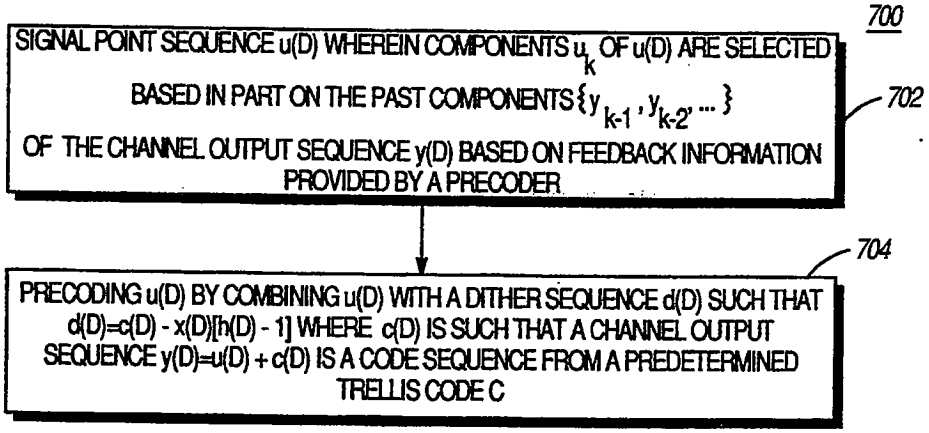


Fig.7

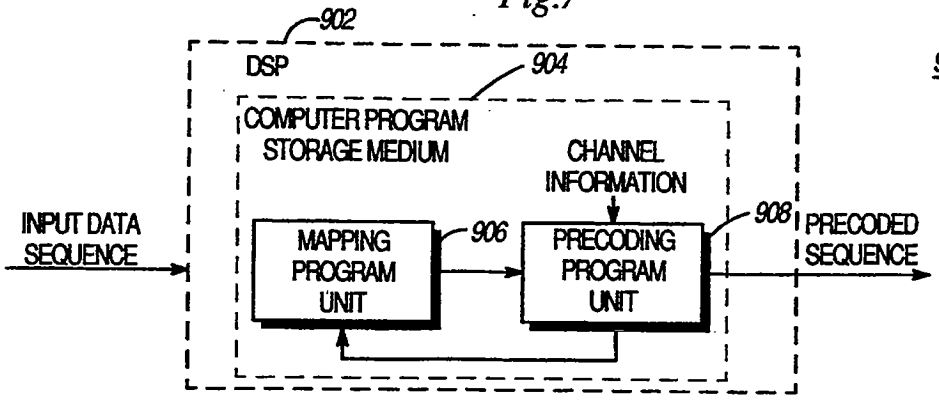


Fig.9

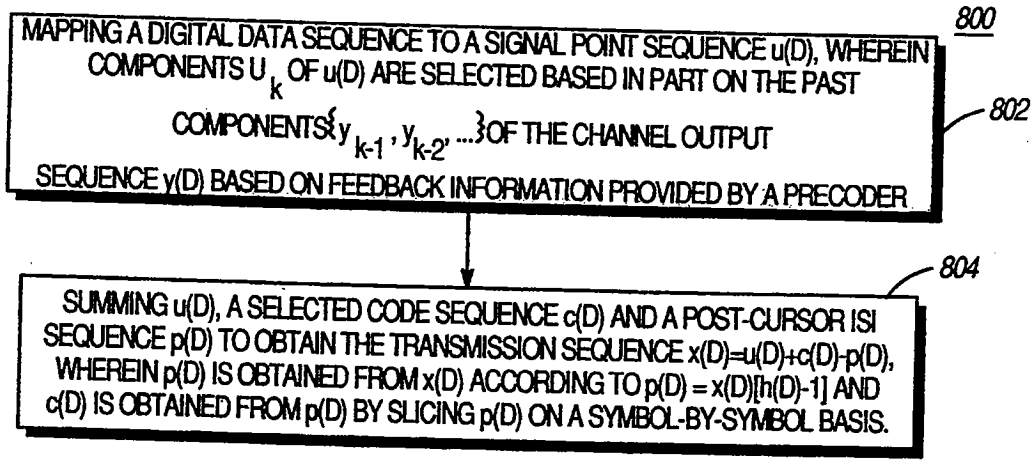


Fig.8

DEVICE AND METHOD FOR PRECODING

FIELD OF THE INVENTION

This invention relates generally to digital communication systems, and more particularly to precoding a digital data sequence for transmission in a digital communication system.

BACKGROUND OF THE INVENTION

It has been shown that on strictly band-limited high-signal-to-noise ratio (SNR) channels with Gaussian noise, digital data may be reliably transmitted at rates approaching channel capacity by using a combination of ideal zero-forcing decision-feedback equalization (DFE) and known coded modulation and constellation shaping techniques designed for ideal channels free of intersymbol interference (ISI). However, ideal DFE is not realizable. Trellis precoding is a realizable combined coding, shaping and equalization technique that achieves the same performance as an ideal DFE along with coding and shaping.

One potential drawback of trellis precoding is that it is effective only for signal constellations whose signal points are uniformly distributed within a space-filling boundary region. Space-filling substantially means that a union of proper non-overlapping translations of the boundary region may cover (tile) the entire space. Stated in another way, the boundary region must be representable as a fundamental region of a lattice, typically referred to as a precoding lattice. To be compatible with known coded modulation techniques, a precoding lattice is typically chosen as a scaled version MZ^2 of a two-dimensional integer lattice Z^2 (where M is a scaling factor) such that the boundary region then has the shape of a square. In certain applications, square signal constellations are not desirable, since they have a higher two-dimensional peak-to-average power ratio (PAR) than constellations with more circular boundaries. More importantly, square constellations are not suitable for representing fractional bits per symbol and require a method known as constellation switching to allow fractional rate transmission, which further increases the two-dimensional PAR. In trellis precoding, it is possible to find precoding lattices whose Voronoi region is more circular than that of a square and which accommodates certain fractional data rates. However, this approach is not very flexible, since it does not uniformly handle all fractional data rates and is more difficult to make invariant to 90° phase rotations, which is an important requirement in certain practical applications. Another drawback of trellis precoding is that to achieve shaping gain, the precoding operation must be combined with shaping operations, which increases the complexity of implementation.

There is a need for a flexible precoding method and device that works with substantially any signal constellation at substantially any data rate and that is implementable independently from constellation shaping while achieving an overall performance that is as close to that of an ideal DFE as possible.

SUMMARY OF THE INVENTION

A device and method are set forth for mapping a digital data sequence into a signal point sequence x(D) for transmission over a channel characterized by a nonideal channel response h(D) using a trellis code C comprising,

a mapper for mapping the digital data sequence into a signal point sequence u(D) such that a component u_k of u(D) at a given time k is selected based in part on past components $\{y_{k-1}, y_{k-2}, \ldots\}$ of a channel output sequence y(D)=x(D)h(D) based on feedback information provided by a precoder, and

a precoder for generating said signal point sequence x(D) according to x(D)=u(D)+d(D), wherein d(D) represents a nonzero difference between a selected non-zero sequence c(D) and a postcursor intersymbol interference (ISI) sequence p(D) substantially of a form p(D)=x(D)[h(D)−1], wherein c(D) is selected such that the channel output sequence y(D) is a code sequence in said trellis code C.

BRIEF DESCRIPTIONS OF THE DRAWINGS

FIG. **1** is a block diagram of a device in accordance with the present invention.

FIG. **2** is a more detailed block diagram illustrating a first embodiment of a device in accordance with the present invention.

FIG. **3** is a detailed block diagram of the mapper of FIG. **2**.

FIG. **4** is a block diagram of a second embodiment of a device in accordance with the present invention.

FIG. **5** is a block diagram of a first embodiment of a digital communication system utilizing a device in accordance with the present invention.

FIG. **6** is a block diagram of a recovery unit of the digital communication system of FIG. **5**, showing the recovery unit with more particularity.

FIG. **7** is a flow diagram setting forth steps of one embodiment in accordance with the method of the present invention.

FIG. **8** is a flow diagram setting forth steps of another embodiment in accordance with the method of the present invention.

FIG. **9** is a block diagram of a digital signal processor used for precoding a digital data sequence to obtain a sequence x(D) for transmission over a discrete-time channel with an impulse response h(D) in accordance with the present invention.

DETAILED DESCRIPTION OF A PREFERRED EMBODIMENT

The method and device of the present invention permits precoding a digital data sequence for transmission over a digital communication channel, performing particularly well on channels having severe attenuation distortion. Substantial benefits are obtained by utilizing the present invention: transmission at substantially any desired data rate without constellation switching, transmission with circular signal constellations, simplification of shaping by completely separating shaping from precoding, and reduction of dither loss by selecting the output of the mapper based in part upon the past components of a channel output sequence based on feedback information provided by a precoder.

As illustrated in FIG.**1** , numeral **100**, a device precodes a digital data sequence in accordance with the present invention to provide a complex precoded sequence $x(D)=x_0+x_1D+\ldots$ for transmission over a discrete-time channel unit with a complex impulse response $h(D)=h_0+h_1D+h_2D^2+\ldots$ using a trellis code

C. In this specification, without losing generality, it is assumed that h(D) is monic (i.e., $h_0=1$).

The code C is a 2n-dimensional trellis code, where n is an integer, based on a lattice partition Λ/Λ', where Λ is a preselected lattice and Λ' is a preselected sublattice of Λ, and a rate m/m+r convolutional code. The lattice Λ is the union of 2^r cosets of a so-called time-zero lattice Λ_0 of the trellis code where Λ_0 is a sublattice of Λ and ζ' is a sublattice of Λ_0. If $y(D)=y_0+y_1D+y_2D^2+\ldots$ is a code sequence in the trellis code, and it is represented as a sequence of 2n-dimensional vector components $y_k=(y_{kn}, \ldots, y_{kn+n-1})$, $k=0, 1, \ldots$, then the component y_k at time k will belong to a unique coset of Λ_0 which is determined by the current state S_k at time k.

The device (100) includes a mapper (102) and a precoder (104). A characteristic of the precoding scheme is that the precoded sequence x(D) may be represented by the sum

$$x(D)=u(D)+d(D)$$

where $u(D)=u_0+u_1D+U_2D^2+\ldots$ is a signal point sequence representing the digital data and is provided by the mapper (102). The signal points $u_i, i=0, 1, 2, \ldots$ are chosen from a two-dimensional signal constellation such that each 2n-dimensional component $U_k=(U_{kn}, \ldots, U_{kn+n-1})$ of U(D) lies on a translate of the lattice Λ. The sequence $d(D)=d_0+d_1D+d_2D^2+\ldots$ is a dither sequence that is generated by the precoder (104) according to

$$d(D)=c(D)-p(D)$$

where $p(D)=x(D)[h(D)-1]$ is a post-cursor intersymbol interference (ISI) sequence and $c(D)=c_0+c_1D+c_2D^2+\ldots$ is chosen such that the channel output sequence $y(D)=x(D)h(D)=u(D)+c(D)$ is a code sequence from the trellis code C.

A feature of the present invention is that the components $c_k=(C_{kn}, \ldots, C_{kn+n-1})$ of the sequence c(D) are selected by the precoder (104) always from the time-zero lattice Λ_0 of the trellis code C or a sublattice Λ_s thereof, and the components u_k of the sequence u(D) are selected by the mapper (102) based in part upon the past values $\{y_{k-1}, y_{k-2}, \ldots\}$ of y(D), such that u_k lies in the coset of Λ_0 in which the component y_k must lie for y(D) to be a valid code sequence in C. The coset is therefore selected based on the state s_k of the code sequence y(D). The digital data determines the signal points from the selected coset as usual. Information about the past values $\{y_{k-1}, y_{k-2}, \ldots\}$ is provided to the mapper (102) by the precoder (104), which is operably coupled to the mapper (102).

The technique of the present invention differs from a technique described in "ISI coder—Combined coding and precoding," AT&T contribution to EIA-TR 30.1, Baltimore, Md., June 1993, in important ways. In the above-referenced technique, the components u_k are always chosen from a translate of the time-zero lattice Λ_0 regardless of the state S_k of y(D), while the components c_k are selected from one of 2^r cosets of Λ_0 depending on the state s_k. The technique of the present invention is logically simpler since the selection of c_k utilizes the same lattice all the time. Furthermore, the method of the present invention may be made transparent to 90 degree rotations (see below), whereas this cannot be achieved completely in the above-referenced technique. Also, in the present invention, the precoder is automatically disabled for ideal channels (i.e., $h(D)=1$), whereas in the above-referenced technique, a special procedure must be followed on ideal channels.

Typically, the present invention is utilized where the complex impulse response h(D) has no zeroes on the unit circle, or equivalently, when its inverse, 1/h(D), is stable. Therefore, the following embodiments utilize an h(D) that is a canonical response with a stable inverse. Note that h(D) may be an all-zero response such as $h(D)=1+0.75\ D$, an all-pole response such as $h(D)=1/(1-0.75\ D)$, or a more general response that includes zeroes and poles. As in earlier precoding techniques, the response h(D) has been determined and is known at the transmitter and the receiver.

FIG. 2, numeral 200, is a more detailed block diagram of a first embodiment of a device in accordance with the present invention. As in FIG. 1, the digital data sequence is first mapped in a mapper (102) into a signal point sequence u(D) using any combination of known encoding, mapping and shaping techniques. The coset of the time-zero lattice Λ_0 in which the components $u_k=(u_{kn}, u_{kn+n-1}, \ldots, u_{kn+n-1})$ lie is determined based on the past values $\{y_{k-1}, y_{k-2}, \ldots\}$ of y(D) based on feedback information provided by the precoder (104). In this embodiment, the precoder comprises a first combining unit (204) and a filtering-/modulo unit (206). The first combining unit (204), typically an adder, is operably coupled to receive the input sequence u(D) and a dither sequence d(D) and forms the precoded sequence $x(D)=u(D)+d(D)$. The filtering/modulo unit (206) includes a filter (212) operably coupled to receive x(D) and is utilized for generating d(D). The filtering/modulo unit (206) further includes a modulo unit (210) that is operably coupled to receive a post-cursor ISI sequence p(D) from the filtering unit (212), where p(D) is obtained by filtering the precoded sequence x(D) that is received from the combining unit (204) in accordance with $p(D)=x(D)[h(D)-1]$. The dither sequence d(D) is given by $d(D)=c(D)-p(D)$, where c(D) is a sequence of 2n-dimensional components c_k chosen from a selected sublattice Λ_s of the time-zero lattice Λ_0. The components c_k are chosen from Λ_s such that the average energy of the dither sequence d(D) is kept small.

A second combining unit (208) (typically an adder) combines the precoded sequence x(D) and the post-cursor ISI sequence p(D) to form the channel output sequence y(D) according to $y(D)=x(D)+p(D)$. The sequence y(D) is then fed back to the mapper for coset selection.

Since the components c_k are chosen from a sublattice Λ_s of Λ_0, the channel output components $y_k=u_k+c_k$ will belong to the same coset of Λ_0 as the signal points u_k. Therefore, the channel output sequence y(D) will be a valid code sequence, provided that u_k's are chosen from an allowable coset of the time-zero lattice based on the current state s_k of y(D)in the trellis code.

The modulo unit (210) in FIG. 2 is utilized for finding the dither components $d_k=(d_{kn}, d_{kn+1}, \ldots, d_{kn+n-1})$ in a specified fundamental region of the sublattice Λ_s that are congruent to the negative post-cursor ISI components $-p_k=(-p_{kn}, -p_{kn+1}, \ldots, -p_{kn+n-1})$ modulo the sublattice Λ_s. The sublattice Λ_s is chosen to obtain a good trade-off between complexity and performance. For low complexity, Λ_s may be selected to be an n-fold Cartesian product of a two-dimensional lattice.

The operation of the modulo unit will now be described in more detail with one specific example. Suppose the trellis code is a four-dimensional trellis code that is based on the lattice partition $RZ^4/2D_4$. This code has the time-zero lattice $\Lambda_0=RD_4$, and therefore Λ_s may be chosen as $2Z^4=(2Z^2)^2$ which is a sublattice of RD_4. In this example, the two-dimensional symbols d_{kn+i}, $i=0, 1, \ldots, n-1$, are chosen on a symbol-by-symbol basis from the square region $[-1, 1)x[-1, 1)$ to be congruent to $-p_{kn+i}$ modulo $2Z^2$ (i.e., the real and imaginary parts of d_{kn+i} are congruent to the real and imaginary parts of $-p_{kn+i}$ modulo 2). In some applications, it is important that the precoding scheme be transparent to 90° phase rotations. This is accomplished by using the interval $[-1, 1)$ for positive components of p_k, and $(-1, 1]$ for non-positive components. In this case, the mapper will also include a differential encoder which is well-known in the state-of-the-art (e.g., L. F. Wei, "Trellis-coded modulation using multi-dimensional constellations," IEEE Trans. Inform. Theory, vol. IT-33, pp. 483–501, July 1987).

The energy of the transmitted symbols $S_x=E\{|x_i|^2\}$ will be the sum S_u+S_d of the energies of u(D) and d(D), where E is a statistical expectation. The average energy S_x of the transmitted sequence x(D) will be approximately the same as the energy S_u of the signal sequence u(D) as long as the average dither energy S_d is small. That means, the better the approximation $c(D)\approx p(D)$ is, the smaller will be the increase in average energy due to the dither sequence d(D). This is achieved by choosing the elements of d_k from a fundamental Voronoi region of the sublattice Λ_s. A key advantage of the present invention is that the dither energy is reduced by modifying the sequence u(D) based on the past history of the channel output sequence y(D).

FIG. 3, numeral **300**, shows a more detailed block diagram of the mapper (**102**) which shows a code tracker (**304**) which is utilized to determine the cosets of Λ_0 from which the components u_k are selected. The input to the code tracker (**304**) is the channel output sequence y(D) obtained from the precoder (**104**). The code tracker (**304**) includes a coset information extractor (**302**) for receiving y(D), a bit mapper (**306**) that is operably coupled to the coset information extractor (**302**) and a rate m/m+r convolutional encoder (**308**) that is operably coupled to the mapper. By tracking the cosets of the sublattice Λ' of the trellis code in which successive components y_k lie, it is possible to keep track of the state of the code sequence y(D). This information is used to determine from which of the 2^r cosets of the time-zero lattice Λ_0 the components u_k should be selected. As shown in FIG. 3, it is possible to implement this step by utilizing the coset information extractor (CIE) (**302**) for first extracting the cosets mentioned above using slicing operations, utilizing the bit mapper (**306**) for converting an output of the CIE to m bits (**304**) and then passing those bits through the rate m/m+r convolutional encoder (**308**) to provide r bits which are used together with the digital data symbols by a mapping unit (**310**) to determine the components u_k. These r bits from the convolutional encoder are used to select one of 2^r cosets of Λ_0 in Λ. For example, in the case of a 4D trellis code based on the partition $RZ^4/2D_4$, a rate m/m+1 convolutional encoder is used to produce one extra bit once every two symbols, and this bit is used to select one of two cosets of the time-zero lattice RD^4 in RZ^4, while the data bits select a four-dimensional point from that coset.

FIG. 4 shows an alternative embodiment, numeral **400**, of the present invention which is substantially equivalent to the first embodiment shown in FIG. 2. In this embodiment, a filtering/modulo unit (**404**) includes a slicing unit (**402**) for first selecting the component c_k from the sublattice Λ_s and the filtering unit (**212**) for forming the dither d_k according to $d_k=c_k-p_k$. In this embodiment, the channel output components y_k may be obtained by combining the component u_k with c_k according to $y_k=u_k+c_k$. Information about the sequence y(D) provided by the precoder is used by the code tracker (**310**) in the mapper (**102**) to determine the coset of the time-zero lattice Λ_0 from which the mapper output u_k is selected.

In another example, suppose that again a four-dimensional trellis code based on the partition $RZ^4/2D_4$ is used, but this time the sublattice Λ_s is the time-zero lattice RD_4 itself. First it should be noted that the sublattice RD_4 may be represented as a union of the 4D lattice $2Z^4$ with its coset $2Z^4+(0, 0, 1, 1)$. Moreover, the 4D lattice $2Z^4$ can be obtained by taking a Cartesian product of the two-dimensional (2D) lattice $2Z^2$ which consists of all pairs of even integers. Therefore RD_4 is represented as

$$RD_4=(2Z^2\times 2Z^2)\,U\,[2Z^2+(1,1)]\,X\,[2Z^2+(1,1)],$$

where U represents the union and X represents the Cartesian product. The union of the 2D lattice $2Z^2$ with its coset $2Z^2+(1,1)$ forms the 2D lattice RZ^2.

Therefore, the slicing unit (**410**) may select the 4D symbols $c_k=(c_{2k}, c_{2k+1})$ by selecting, in the even symbol interval 2k, its symbol c_{2k} from RZ^2. If c_{2k} belongs to $2Z^2$ then in the following odd symbol interval, the second symbol c_{2k+1} is selected from the even integer lattice $2Z^2$. If c_{2k} belongs to the coset $2Z^2+(1,1)$, however, then in the next odd symbol interval, the second symbol c_{2k+1} is selected from the coset $2Z^2+(1,1)$. This way it is ensured that the 4D symbol (c_{2k}, c_{2k+1}) will belong to RD_4.

It should be noted that the invention is not limited to criteria that minimize the average dither energy, and any criterion may be used to select the code sequence c(D) as long as the selection of each c_i is based only upon past values x_j, $j<i$, of x(D), and the components c_k belong to the time-zero lattice Λ_0. For example, in certain applications it may be desirable to limit the range of the channel output symbols $y_i=u_i+c_i$. This may be achieved, at the expense of a higher dither energy S_d, by restricting the values of c_i to a certain range.

The above description utilizes an assumption that the channel is characterized by a discrete-time complex impulse response h(D). It is well-known in the state-of-the-art that any discrete time or continuous-time linear channel with additive noise may be represented by a canonical discrete-time equivalent channel with a causal ($h_k=0$, $k<0$), minimum-phase (all zeros outside or on the unit circle), monic ($h_0=1$) impulse response h(D) and additive white noise w(D). A canonical receiver front-end that includes a whitened matched filter and a sampler (in the case of continuous-time channels) operating at a selected symbol rate may be utilized to provide such an equivalent channel. It should be mentioned that in practice, typically, h(D) represents the combined effect of the filters in the transmitter, channel, the receiver, and a sampler. Similarly, w(D) represents the noise after it passes through the receive filters and the sampler. The whitened-matched filter reduces the

strength of the distortion through proper filtering and therein lies the performance advantage of the present invention over conventional linear equalizations.

In practice, when h(D) is an all-zero response, a whitened matched filter may be determined adaptively using standard adaption techniques for decision-feedback equalizers. When it is desired that h(D) be an all-pole filter, then one first determines adaptively an all-zero response h'(D) using the standard methods and then finds $h(D)=1/g'(D)$ using well-known polynomial division techniques, where g'(D) is a finite polynomial approximately equal to $g'(D)\approx 1/h'(D)$.

A first embodiment of a device of the present invention incorporated into a digital communication system is illustrated in the block diagrams of FIG. 5, numeral **500**, wherein at least one of a transmission unit and a receiving unit utilizes the present invention. The said system typically includes at least one of a transmission unit (**502**) and a receiving unit (**506**) wherein the transmission unit has a mapper (**508**) and a precoder (**510**) for transmitting a digital data sequence and a channel (**504**) obtained as described in the above paragraph, operably coupled to the precoder (**510**), for facilitating transmission of the precoded sequence x(D), and the receiving unit (**506**) has a decoding unit (**518**), operably coupled to the channel unit (**504**), for receiving and decoding a received sequence r(D) to provide an estimated output sequence $\hat{y}(D)$, and a recovery unit (**520**), operably coupled to the decoding unit (**518**), for substantially recovering an estimate $\hat{u}(D)$ of the signal point sequence u(D). An estimate of the transmitted digital data sequence is then found from $\hat{u}(D)$ using an inverse map and shaping recovery (if constellation shaping is employed).

The equivalent channel (**504**), represented as set forth above, is substantially represented by a filter (**512**) having a response h(D), for receiving x(D) and producing an output sequence $y(D)=x(D)h(D)$, defined earlier, an additive noise unit (**516**) for providing additive noise, and a combining unit (**514**), typically a summer, operably coupled to the channel filter h(D) (**512**) and to the additive noise unit (**516**).

The decoding unit (**518**) is typically a decoder for the trellis code C, as is known in the art. The decoding unit (**518**), typically receives and decodes a noisy received sequence r(D) which is of a form:

$$\begin{aligned} r(D) &= x(D)h(D) + w(D) \\ &= y(D) + w(D) \\ &= [u(D) + c(D)] + w(D), \end{aligned}$$

to provide an estimate $\hat{y}(D)$ of the channel output sequence $y(D)=x(D)h(D)$, and a recovery unit (**520**), operably coupled to the decoding unit (**518**), substantially recovers an estimate $\hat{u}(D)$ of the input sequence u(D), described more fully below.

As described earlier, the sequence y(D) must be a sequence in the trellis code C. That means that the sequence y(D) may be estimated by a conventional decoder for C, as is known in the art, to provide an estimated output sequence $\hat{y}(D)$.

The recovery unit (**520**), illustrated with more particularity in the block diagram of FIG. 6, numeral **600**, typically includes at least a recovery filtering unit (**604**) substantially the same as the filter (**212**) in the precoder (**104**), operably coupled to receive an estimated sequence $\hat{y}(D)$, for filtering $\hat{y}(D)$ to obtain an estimate $\hat{p}(D)$ of the post-cursor ISI sequence p(D), substantially of a form $\hat{p}(D)=\hat{y}(D)/\{1-1/h(D)\}$ and for providing $\hat{p}(D)$ as a feedback signal to obtain $\hat{x}(D)$ from $\hat{y}(D)$ according to $\hat{x}(D)=\hat{y}(D)-\hat{p}(D)$, a recovery modulo unit (**606**), operably coupled to the recovery filtering unit (**604**), for finding the estimated dither sequence $\hat{d}(D)$, as the sequence whose components $\hat{d}_k=(\hat{d}_{kn}, \hat{d}_{kn+1}, \ldots, \hat{d}_{kn+n-1})$ lie in a specified fundamental region of the sublattice Λ_s, and are congruent to the negative post-cursor ISI components $-\hat{p}_k=(-\hat{p}_{kn}, -\hat{p}_{kn+1}, \ldots, -\hat{p}_{kn+n-1})$ modulo the sublattice Λ_s in a manner that is substantially the same as that used in the precoding unit at the transmitter, and a recovery combining unit (**606**), operably coupled to the recovery modulo unit (**604**) and to the estimator combiner (**602**) for receiving the estimated sequence $\hat{x}(D)$ of the precoded sequence x(D), for substantially determining a difference between the sequence $\hat{x}(D)$ and the sequence $\hat{d}(D)$ to obtain the estimate $\hat{u}(D)$ of the original input sequence u(D). As long as there are no decision errors ($\hat{y}(D)=y(D)$),and the operations in the transmitter and receiver are substantially symmetrical, the original sequence u(k) will be correctly recovered. Other equivalent implementations of the recovery circuit are also possible.

To summarize, the recovery filtering unit (**604**) is utilized to reconstruct an estimate $\hat{p}(D)$ of a post-cursor ISI variable p(D), then the recovery modulo unit (**604**) is utilized to determine a dither sequence $\hat{d}(D)$ that substantially correlates with d(D) in the corresponding device (**100**) at the transmitter, and then utilizes the recovery combining unit (**608**) to provide $\hat{u}(D)=\hat{x}(D)-\hat{d}(D)$.

Of course, there will be occasional errors in $\hat{y}(D)$ due to channel noise, and these may lead to error propagation. However, since 1/h(D) is stable, the error propagation in the filter $1-1/h(D)$ will never be catastrophic. Moreover, if h(D) is an all-pole response of order p (where m is a selected integer), then error propagation will be strictly limited to at most p symbols.

Where an estimate $\hat{u}_i$, an i'th variable of the recovered sequence u(D), falls outside the allowed range of a i'th variable u_i of the input sequence u(D), such a range violation indicates that a decision error has occurred either in the current symbol y_i, or c_i is in error because of an error in some recent symbol y_{j-i}, $i>0$. When such range violations are detected, one may try to correct the violations by adjusting the estimates y_i or y_{j-i}. Thus, by monitoring the range violations, some degree of error correction is achieved. Such an error detection capability may also be useful for monitoring the performance of the transmission system.

Thus, a digital communications receiver may be utilized in accordance with the present invention for receiving a digital data sequence that was mapped into a precoded sequence x(D) and transmitted over a channel characterized by a nonideal response h(D) using a trellis code C, providing a received sequence r(D), where a receiver includes at least a decoding unit, operably coupled to receive r(D), for decoding the received transmission sequence r(D) to provide an estimated output sequence $\hat{y}(D)$, and a recovery unit, operably coupled to the decoding unit, for substantially recovering an estimated sequence $\hat{u}(D)$ for a sequence u(D) for a transmitted signal point sequence x(D) which is generated according to $x(D)=u(D)+d(D)$, wherein u(D) is a signal point sequence which uniquely represents said digital data sequence and wherein the coset of the time-

zero lattice Λ_0 in which 2n-dimensional components u_k lie depends on the state of the channel output sequence y(D), and d(D) represents a nonzero difference between a selected sequence c(D) whose components c_k are selected from a sublattice Λ_s of the time-zero lattice Λ_0 of the trellis code and a post-cursor intersymbol interference (ISI) sequence p(D) substantially of a form $p(D)=x(D)[h(D)-1]$, such that c(D) is selected based only upon p(D).

As illustrated in FIG. 6, one embodiment of the recovery means utilizes an estimator combining unit (602) that is operably coupled to receive the estimated output sequence $\hat{y}(D)$ and $\hat{p}(D)$, a recovery filtering unit (604), operably coupled to receive the estimated output sequence $\hat{x}(D)$, for providing an estimated post-cursor intersymbol interference (ISI) sequence $\hat{p}(D)$, a recovery modulo unit (606), operably coupled to the recovery filtering means, for providing an estimated dither sequence $\hat{d}(D)$ that substantially correlates with d(D) utilized for providing the transmission sequence x(D), a third combining unit (610) (typically a summer), operably coupled to receive the estimated precoded sequence $\hat{x}(D)$ and to the output $\hat{d}(D)$ of the recovery modulo means, for determining the estimated sequence $\hat{u}(D)$, substantially of a form $\hat{u}(D)=\hat{x}(D)-\hat{d}(D)$, and an inverse mapper (612), operably coupled to the third combining means, for inverse mapping the estimated sequence $\hat{u}(D)$ to provide a recovered digital data sequence substantially equal to the transmitted digital data sequence.

In addition the digital communications receiver may be selected such that the decoding unit (518) further includes a reduced complexity sequence estimator unit that utilizes a correlation between successive symbols y_i. In one implementation, the reduced complexity sequence estimator unit utilizes a sequence estimator having a reduced number of states that are determined utilizing state merging techniques for reduced-state sequence estimation (RSSE).

Where desired, the recovery unit may be selected to include a range violation determiner unit. When an i'th variable $\hat{u}_i$ of the recovered sequence $\hat{u}(D)$ is outside a certain range (a range violation), this unit adjusts at least one of an estimate $\hat{y}_i$ and a past estimate $\hat{y}_{k-j}$ (where j is a positive integer) to substantially correct the range violation.

FIG. 7, numeral 700, sets forth a flow chart illustrating steps in accordance with the method of the present invention for precoding a stream of signal points for transmission in a digital communication system. The method provides for precoding a digital data sequence to generate a sequence x(D) for transmission over a discrete-time channel with an impulse response h(D) using a trellis code C. A stream of signal points u(D) is transmitted as $x(D)=u(D)+d(D)$, where d(D) is a dither sequence of a form $d(D)=c(D)-p(D)$, where $p(D)=x(D)[h(D)-1]$represents a post-cursor intersymbol interference (ISI), and c(D) is a sequence whose components c_k belong to a sublattice $\Lambda_s$of the time-zero lattice Λ_0 of a trellis code C, and c(D) is obtained based only upon p(D). The sequence u(D) uniquely represents the digital data sequence, and its components u_k are selected based in part on past components $\{y_{k-1}, y_{k-2}, \ldots\}$ of the channel output sequence $y(D)=x(D)h(D)$, based on feedback information provided by the precoder, such that y(D) is a code sequence in the trellis code.

In one embodiment, illustrated in FIG. 7, numeral 700, the method for mapping a digital data sequence into a signal point sequence x(D) for transmission over a channel characterized by a nonideal channel response h(D) using a trellis code C includes the steps of: (1) mapping the digital data sequence to a signal point sequence u(D) (702), and (2) precoding the signal point sequence u(D) by combining u(D) with a dither sequence d(D) (704) such that d(D) is of a form: $d(D)=c(D)-x(D)[h(D)-1]$, where c(D) is selected such that a channel output sequence $y(D)=u(D)+c(D)$ is a code sequence from the trellis code C. The components u_k of u(D) are selected based in part on past components $\{y_{k-1}, y_{k-2}, \ldots\}$ of the channel output sequence $y(D)=x(D)h(D)$. The components c_k of c(D) are selected from a sublattice $\Lambda_s$of the time-zero lattice Λ_0 of the trellis code. The sequences are as described above.

In another embodiment, illustrated in FIG. 8, numeral 800, the method comprises the steps of mapping the digital data sequence to a signal point sequence u(D) (802), summing u(D), a selected code sequence c(D) and a post-cursor ISI sequence p(D) to obtain the transmission sequence $x(D)=u(D)+c(D)-p(D)$ (804), filtering x(D) to obtain (806) p(D) substantially of a form:

$$p(D)=x(D)[h(D)-1],$$

and slicing p(D) on a symbol-by-symbol basis to obtain the sequence c(D). Further modifications of the method may be utilized in accordance with the modifications described more fully above for the device of the present invention. Again, the components u_k of u(D) are selected based in part on past components $\{y_{k-1}, y_{k-2}, \ldots\}$ of the channel output sequence $y(D)=x(D)h(D)$. The components c_k of c(D) are selected from a sublattice Λ_s of the time-zero lattice Λ_0 of the trellis code.

The present invention may be implemented in a digital communication system, illustrated in FIG. 9, numeral 900, where a digital signal processor (902) is utilized to precode a digital data sequence to obtain a sequence x(D) for transmission over a discrete-time channel with an impulse response h(D). The processor typically includes a program storage medium (904) having a computer program to be executed by the digital signal processor, the program including a mapping program (906) for mapping the digital data sequence into a signal point sequence u(D) based in part upon the past values of a channel output sequence y(D) based on information provided by a precoding program and a precoding program (908) for utilizing the u(D) to generate a sequence x(D) wherein x(D) may be represented as the sum $u(D)+d(D)$ of the stream of signal points u(D) which uniquely represents the digital data sequence and is also chosen based on the state of channel output sequence y(D) and a dither sequence $d(D)=c(D)-p(D)$, where c(D) is a sequence from a sublattice Λ_s of the time-zero lattice Λ_0 of the trellis code C and where p(D) represents a post-cursor intersymbol interference (ISI) sequence of a form $p(D)=x(D)\{h(D)-1\}$. The code sequence c(D) is determined based upon only the post-cursor ISI sequence p(D). The components u_k of u(D) are selected based in part on past components $\{y_{k-1}, y_{k-2}, \ldots\}$ of the channel output sequence $y(D)=x(D)h(D)$. The components c_k of c(D) are selected from a sublattice Λ_s of the time-zero lattice Λ_0 of the trellis code. Further description of the operation of the processor follows that described above.

The processor typically includes a computer program storage medium having a computer program to be executed by the digital signal processor where the computer program includes a mapping program for mapping the digital data sequence into a signal point sequence u(D) based in part upon the past values of a channel output sequence y(D) based on feedback information provided by a precoding program, and a precoding program for selecting said signal point sequence x(D) from a subset of all possible signal point sequences that are of a form $x(D)=u(D)+d(D)$, wherein d(D) represents a nonzero difference between a selected nonzero sequence c(D) and a postcursor intersymbol interference (ISI) sequence p(D) substantially of a form $p(D)=x(D)[h(D)-1]$, where c(D) is selected based upon p(D) such that the channel output sequence $y(D)=u(D)+c(D)$ is a code sequence from a translate of said trellis code C. In one embodiment, the precoding program includes first combining instruction(s) for combining the input sequence u(D) and a dither sequence d(D) to generate the precoded sequence $x(D)=u(D)+d(D)$, d(D) generating/p(D) generating instructions for generating d(D) and for providing a post-cursor ISI sequence p(D), second combining instructions for generating a sequence $y(D)=x(D)+p(D)$.

The present invention relies on past channel output signals to remove a dither sequence d(D) that is added to an input sequence u(D) at the transmitter to form a transmitted sequence, $x(D)=u(D)+d(D)$, the dither sequence being substantially a difference between a post-cursor intersymbol interference p(D) and an appropriate sequence, c(D), from a sublattice Λ_s of the time-zero lattice Λ_0 of a trellis code. The sequence u(D) is chosen based in part on past components $\{y_{k-1}, y_{k-2}, \ldots\}$ of the channel output sequence $y(D)=x(D)h(D)$ based on feedback information provided by the precoder. The coset of the time-zero lattice Λ_0 in which successive elements of u(D) should lie are determined based on the past values $\{y_{k-1}, y_{k-2}, \ldots\}$.

The present invention may be utilized with virtually any signaling method and at any data rate. Further, the present invention may be utilized independently of constellation shaping techniques (e.g., shell mapping, "Signal mapping and shaping for V.fast," Motorola contribution D196, CCITT Study Group XVIII, Geneva, June 1992); that means u(D) may represent an already shaped sequence whose signal points have a nonuniform Gaussian-like probability distribution.

In the present invention, the dither sequence may increase the average transmit energy. Since in practice, the average transmit energy must be kept constant, the signal x(D) must be scaled down to maintain the same average energy. The increase in the average transmit energy is referred to herein as a dithering loss.

Although several exemplary embodiments are described above, it will be obvious to those skilled in the art that many alterations and modifications may be made without departing from the invention. For example, even though primarily trellis codes are described above, the method may be used with block or lattice codes as well. It may also be used with selected multilevel trellis codes. Also, although two-dimensional (passband, quadrature) transmission systems are emphasized, the methods may also be applied to one-dimensional (baseband) or higher-dimensional (parallel channels) transmission systems. Further, the invention may be utilized with trellis codes whose dimensionality is odd. Although the description above emphasizes channel responses h(D) that are monic, the invention may also be applied to more general channel responses with $h_0 \neq 1$, by either scaling the channel response to make it monic, or by appropriately scaling the variables of the precoding system. All such implementations are considered substantially equivalent to the present invention.

Accordingly, it is intended that all such alterations and modifications be included within the spirit and scope of the invention as defined in the appended claims.

I claim:

1. A device for mapping an input digital data sequence into an output signal point sequence x(D) for transmission over a channel characterized by a nonideal channel response h(D) using a trellis code C comprising:

a mapper for mapping the digital data sequence into a signal point sequence u(D) such that the components u_k of u(D), where k is a time index, are selected based in part on past components $\{y_{k-1}, y_{k-2}, \ldots\}$ of a channel output sequence $y(D)=x(D)h(D)$ which are obtained based on feedback information provided by a precoder,

a precoder for generating said output signal point sequence x(D) according to $x(D)=u(D)+d(D)$, wherein d(D) represents a nonzero difference between a selected sequence c(D) and a postcursor intersymbol interference (ISI) sequence p(D) substantially of a form $p(D)=x(D)[h(D)-1]$, wherein c(D)is selected such that the channel output sequence y(D) is a code sequence in said trellis code C.

2. The device of claim **1** wherein the components c_k of c(D) are selected from a time-zero lattice Λ_0 of said trellis code or a sublattice $\Lambda_s$thereof.

3. The device of claim **1** wherein said components u_k of u(D) are selected based on the state S_k of the channel output sequence y(D) in said trellis code.

4. The device of claim **3** wherein said components u_k of u(D) at time k are selected from one of the cosets of the time-zero lattice Λ_0 based on the state s_k of the channel output sequence y(D) in said trellis code.

5. The device of claim **4** wherein said cosets of the time-zero lattice are determined by utilizing a convolutional encoder for the said trellis encoder.

6. The device of claim **1** wherein said trellis code is a four-dimensional trellis code.

7. The device of claim **6** wherein the trellis code is based on the lattice partition $RZ^4/2D_4$ and its time-zero lattice is RD_4.

8. The device of claim **7**, wherein said mapper further includes a differential encoder.

9. The device of claim **8**, wherein said mapper further includes constellation shaping.

10. The device of claim **9**, wherein said constellation shaping is achieved using shell mapping.

11. The device of claim **6** wherein a slicing means provides the selected sequence c(D) by selecting the symbols c_{2k} and c_{2k+1} from the integer lattice $2Z^2$.

12. The device of claim **6** wherein a slicing means provides the selected sequence c(D) by selecting a symbol C_{2k} from RZ^2 in a first symbol interval, a symbol c_{2k+1} from either $2Z^2$ or its coset $2Z^2+(1,0)$ in a second symbol interval, based on c_{2k}, where k is a time index.

13. The device of claim 6 wherein, the selection of c_k further includes constraints to limit a range of the symbols y_i of the channel output sequence y(D).

14. The device of claim 1 wherein the precoder comprises at least:

first combining means, operably coupled to the mapper, and to a modulo means, for combining u(D) and at least the dither sequence d(D) to provide a precoded sequence x(D),

wherein the modulo means is operably coupled to a filter, and is utilized for finding the dither sequence d(D) based on the post-cursor ISI sequence p(D),

wherein the filter is operably coupled to the first combining means, and is utilized for extracting the post-cursor ISI sequence p(D) from the transmission sequence x(D).

15. The device of claim 14 wherein the first combining means is a summer.

16. The device of claim 14 wherein the precoder further includes second combining means, operably coupled to the first combining means and to the filter, for obtaining the channel output sequence y(D) substantially of a form $y(D)=x(D)+p(D)$.

17. The device of claim 1 wherein the precoding unit comprises at least:

first combining means, operably coupled to the mapping means and to a slicing means, for combining u(D) and at least the sequence c(D) to provide the channel output sequence y(D),

wherein the slicing means is operably coupled to a filter, and is utilized for slicing the post-cursor sequence p(D) to a sequence of signal points c(D) whose components C_k are selected from a time-zero lattice Λ_0 of the trellis code C or a sublattice Λ_s thereof,

a second combining means, operably coupled to the first combining means and the filter, for combining the channel output sequence y(D) and the post-cursor ISI sequence p(D) to form the precoded sequence x(D), and

wherein the filter is operably coupled to the second combining means, and is utilized for extracting the post-cursor ISI sequence p(D) from the transmission sequence x(D).

18. The device of claim 17 wherein the first combining means is a summer.

19. The device of claim 17 wherein the second combining means is a summer.

20. The device of claim 1 wherein the trellis code C is a block code.

21. A digital communications receiver for receiving a digital data sequence that was mapped into a signal point sequence x(D) and transmitted over a channel characterized by a nonideal response h(D) using a trellis code C, providing a received sequence r(D), comprising at least:

decoding means, operably coupled to receive r(D), for decoding the received transmission sequence r(D) to provide an estimated output sequence $\hat{y}(D)$, and

recovery means, operably coupled to the decoding means, for substantially recovering an estimated sequence $\hat{u}(D)$ for a sequence u(D) representing said digital data sequence by first recovering an estimate of the transmitted signal point sequence x(D) generated by a precoder according to $x(D)=u(D)+d(D)$, wherein d(D) represents a nonzero difference between a selected non-zero sequence c(D) and a postcursor intersymbol interference (ISI) sequence p(D) substantially of a form $p(D)=x(D)[h(D)-1]$, wherein c(D) is selected such that the channel output sequence $y(D)=x(D)h(D)$ is a code sequence in said trellis code C and u(D) is selected such that a component u_k at a given time k is selected based in part on past components $\{y_{k-1}, y_{k-2}, \ldots\}$ of a channel output sequence y(D) based on feedback information provided by the precoder, where $u(D)=u_0+u_1D+u_2D^2+, \ldots$ is a signal point sequence representing the digital data provided by a mapper and transmitted to the digital communications receiver.

22. The digital communications receiver of claim 21 wherein the recovery means includes at least:

recovery filtering means, operably coupled to receive the estimated output sequence $\hat{y}(D)$, for providing an estimated post-cursor intersymbol interference (ISI) sequence $\hat{p}(D)$,

recovery slicing means, operably coupled to the recovery filtering means, for providing an estimated nonzero dither sequence $\hat{d}(D)$ that substantially correlates with d(D) utilized for providing the transmission sequence x(D),

third combining means, operably coupled to receive the estimated output sequence $\hat{y}(D)$ and to the recovery slicing means, for determining the estimated sequence $\hat{u}(D)$, substantially of a form $\hat{u}(D)=\hat{y}(D)-\hat{p}(D)-\hat{d}(D)$, and

an inverse mapping means, operably coupled to the third combining means, for inverse mapping the estimated sequence $\hat{u}(D)$ to provide a recovered digital data sequence.

23. The digital communications receiver of claim 21 wherein the nonideal response h(D) represents the impulse response of a noise prediction filter.

24. The digital communications receiver of claim 21 wherein the decoding means further includes a reduced complexity sequence estimator means that utilizes a correlation between successive symbols y_i.

25. The digital communications receiver of claim 24 wherein the reduced complexity sequence estimator means utilizes a sequence estimator having a reduced number of states that are determined utilizing state merging techniques for reduced-state sequence estimation (RSSE).

26. A method for mapping an input digital data sequence into an output signal point sequence x(D) for transmission over a channel characterized by a nonideal channel response h(D) using a trellis code C comprising the steps of:

mapping the digital data sequence into a signal point sequence u(D) such that a component u_k of u(D) at a given time k is selected based in part on past components $\{y_{k-1}, y_{k-2}, \ldots\}$ of a channel output sequence $y(D)=x(D)h(D)$ based on feedback information provided by a precoder,

generating, by the precoder, said signal point sequence x(D) according to $x(D)=u(D)+d(D)$, wherein d(D) represents a nonzero difference between a selected non-zero sequence c(D) and a postcursor intersymbol interference (ISI) sequence p(D) substantially of a form $p(D)=x(D)[h(D)-1]$, wherein c(D) is selected such that the channel output sequence y(D) is a code sequence in said trellis code C.

27. The method of claim 26 wherein the components C_k of c(D) are selected from a time-zero lattice Λ_0 of said trellis code or a sublattice Λ_s thereof.

28. The method of claim 26 wherein said components u_k of u(D) are selected based on the state s_k of the channel output sequence y(D) in said trellis code.

29. The method of claim 28 wherein said components u_k of u(D) at time k are selected from one of the cosets of the time-zero lattice Λ_0 based on the state S_k of the channel output sequence y(D) in said trellis code.

30. The method of claim 29 wherein said cosets of the time-zero lattice are determined by utilizing a convolutional encoder for the said trellis encoder.

31. The method of claim 26 wherein said trellis code is a four-dimensional trellis code.

32. The method of claim 31 wherein the trellis code is based on the lattice partition $RZ^4/2D_4$ and its time-zero lattice is RD_4.

33. The method of claim 32, wherein said mapper further includes a differential encoder.

34. The method of claim 33, wherein said mapper further includes constellation shaping.

35. The method of claim 34, wherein said constellation shaping is achieved using shell mapping.

36. A digital signal processor having at least a plurality of registers and an arithmetic logic unit, for use in a digital communication system to precode a digital data sequence into a signal point sequence x(D) for transmission over a discrete-time channel with a impulse response h(D) using a trellis code C, the processor having a device comprising:

mapping means for mapping, using the plurality of registers and the arithmetic logic unit, the digital data sequence into a signal point sequence u(D) such that a component u_k of u(D) at a given time k is selected based in part on past components $\{y_{k-1}, y_{k-2}, \ldots\}$ of a channel output sequence y(D)=x(D)h(D) based on feedback information,

precoding means for generating, using the plurality of registers and the arithmetic logic unit, said signal point sequence x(D) according to x(D)=u(D)+d(D), wherein d(D) represents a nonzero difference between a selected non-zero sequence c(D) and a postcursor intersymbol interference (ISI) sequence p(D) substantially of a form p(D)=x(D)[h(D)−1], wherein c(D) is selected such that the channel output sequence y(D) is a code sequence in said trellis code C.

37. The signal processor of claim 36 wherein the components c_k of c(D) are selected from a time-zero lattice Λ_0 of said trellis code or a sublattice Λ_s thereof.

38. The signal processor of claim 36 wherein said components u_k of u(D) are selected based on the state s_k of the channel output sequence y(D) in said trellis code.

39. A digital communication system for precoding a digital data sequence into a signal point sequence x(D) for transmission over a discrete-time channel with a impulse response h(D) using a trellis code C, comprising at least one of:

a transmission unit and

a receiving unit,

wherein the transmission unit having a digital signal processor with at least a plurality of registers and an arithmetic logic unit, comprises:

a mapper for mapping, using the plurality of registers and the arithmetic logic unit, the digital data sequence into a signal point sequence u(D) such that a component u_k of u(D) at a given time k is selected based in part on past components $\{y_{k-1}, y_{k-2}, \ldots\}$ of a channel output sequence y(D)=x(D)h(D) based on feedback information provided by a precoder,

a precoder for generating, using the plurality of registers and the arithmetic logic unit, said signal point sequence x(D) according to x(D)=u(D)+d(D), wherein d(D) represents a nonzero difference between a selected non-zero sequence c(D) and a postcursor intersymbol interference (ISI) sequence p(D) substantially of a form p(D)=x(D)[h(D)−1], wherein c(D) is selected such that the channel output sequence y(D) is a code sequence in said trellis code C,

and the receiving unit includes:

a decoding unit, operably coupled to the channel, for receiving and decoding a received sequence r(D) to provide an estimated output sequence $\hat{y}(D)$, and

a recovery unit, operably coupled to the decoding unit, for substantially recovering an estimate $\hat{u}(D)$ of the signal point sequence u(D).

40. The system of claim 39 wherein the decoding unit is a decoder for the trellis code C that receives and decodes a noisy received sequence r(D) which is of a form:

$$\begin{aligned} r(D) &= x(D)h(D) + w(D) \\ &= y(D) + w(D) \\ &= [u(D) + c(D)] + w(D), \end{aligned}$$

where w(D) represents additive white noise, to provide an estimate $\hat{y}(D)$ of the channel output sequence y(D)=x(D)h(D), and the recovery unit, operably coupled to the decoding unit, substantially recovers an estimate $\hat{u}(D)$ of the input sequence u(D).

41. The system of claim 39 wherein the components C_k of c(D) are selected from a time-zero lattice Λ_0 of said trellis code or a sublattice Λ_s thereof.

42. The device of claims 39 wherein said components u_k of u(D) are selected based on the state S_k of the channel output sequence y(D) in said trellis code.

* * * * *

EXHIBIT D

U 7042968

THE UNITED STATES OF AMERICA

TO ALL TO WHOM THESE PRESENTS SHALL COME:

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office

December 08, 2006

THIS IS TO CERTIFY THAT ANNEXED HERETO IS A TRUE COPY FROM THE RECORDS OF THIS OFFICE OF:

U.S. PATENT: *6,198,776*
ISSUE DATE: *March 06, 2001*

By Authority of the
Under Secretary of Commerce for Intellectual Property
and Director of the United States Patent and Trademark Office

M. K. CARTER
Certifying Officer

GE 000043

US006198776B1

(12) **United States Patent**
Eyuboglu et al.

(10) **Patent No.: US 6,198,776 B1**
(45) **Date of Patent: *Mar. 6, 2001**

(54) **DEVICE AND METHOD FOR PRECODING DATA SIGNALS FOR PCM TRANSMISSION**

(75) Inventors: **M. Vedat Eyuboglu**, Concord; **Pierre A. Humblet**, Cambridge; **Dae-young Kim**, Lexington, all of MA (US)

(73) Assignee: **Motorola Inc.**, Schaumburg, IL (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

This patent is subject to a terminal disclaimer.

(21) Appl. No.: **08/999,249**

(22) Filed: **Dec. 29, 1997**

Related U.S. Application Data

(63) Continuation-in-part of application No. 08/747,840, filed on Nov. 13, 1996, now Pat. No. 5,818,075.

(51) **Int. Cl.**[7] **H04L 25/49**
(52) **U.S. Cl.** **375/286**; 375/295
(58) **Field of Search** 375/286, 216, 375/242, 222, 295; 379/93.01

(56) **References Cited**

U.S. PATENT DOCUMENTS

5,394,437	2/1995	Ayanoglu et al.	375/222
5,528,625	6/1996	Ayanoglu et al.	375/222
5,659,579	8/1997	Herzberg	375/262
5,825,816 *	10/1998	Cole et al.	375/222

FOREIGN PATENT DOCUMENTS

01657 10/1997 (FR) .

OTHER PUBLICATIONS

Dagdeviren, Nuri; "Proposed Baseline For PCM Upstream," Document Number: TR–30.Dec. 1996. Lucent Technologies; Midletown, NJ. Email: dagdeviren @lucent.com. Dec. 4–5, 1996.

Eyuboglu, Vedat; "Generalized Spectral Shaping for PCM Modems," Telecommunications Industry Association, TR30.1 Meeting, Norcross, Georgia, Apr. 9–11, 1997. pp. 1–5.

Eyuboglu, Vedat; "Convolutional Spectral Shaping," Telecommunications Industry Association, TR30.1 Meeting, Norcross, Georgia, Apr. 9–11, 1997. pp. 1–4.

Eyuboglu, Vedat; "More on Convolutional Spectral Shaping," ITUTelecommunications Standardization Sector 009, V.pcm Rapporteur Meeting, La Jolla, CA, May 5–7, 1997, pp. 1–5.

Eyuboglu, Vedat; "Draft Text for Convolutional Spectral Shaping," ITU–T SG 16 Q23 Rapporteur's Meeting, Sep. 2–11, 1997, Sun River, Oregon; p. 1–2.

Eyuboglu, Vedat; "A Comparison of CSS and Maximum Inversion," Telecommunications Industry Association, TR30.1 Meeting on PCM Modems, Galveston, Texas, Oct. 14–16, 1997; 6 pages.

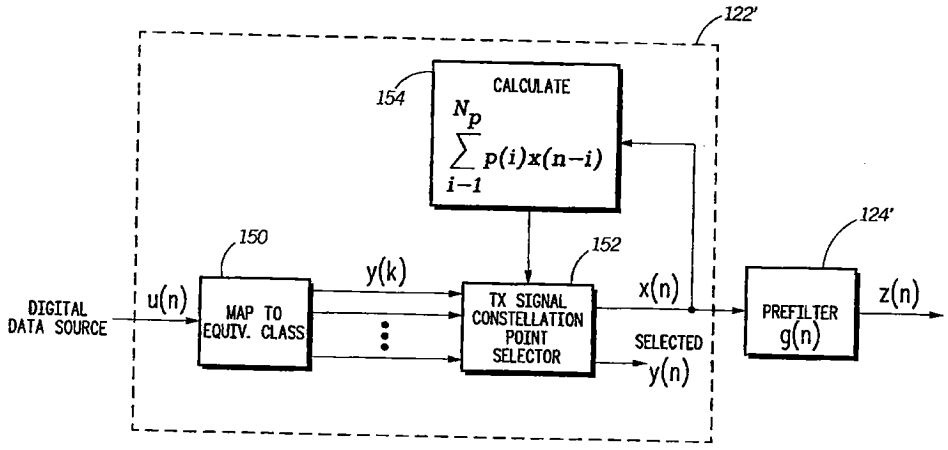
(List continued on next page.)

Primary Examiner—Temesghen Ghebretinsae
(74) *Attorney, Agent, or Firm*—Joanne N. Pappas; John W. Powell

(57) **ABSTRACT**

A device and method for preceding data signals for pulse code modulation (PCM) transmission includes a transmitter for transmitting a sequence of analog levels over an analog channel to a quantization device, wherein the analog channel modifies the transmitted analog levels, the transmitter comprising: a mapping device for mapping data bits to be transmitted to a sequence of equivalence classes, wherein each equivalence class contains one or more constellation points; and a constellation point selector interconnected to the mapping device which selects a constellation point in each equivalence class to represent the data bits to be transmitted and which transmits an analog level that produces the selected constellation point at an input to the quantization device.

31 Claims, 10 Drawing Sheets



OTHER PUBLICATIONS

Eyuboglu, Vedat; "Draft Text for Convolutional Spectral Shaping," Telecommunications Industry Association, TR30.1 Meeting Galveston, Texas, Oct. 14–16, 1997; 2 page.

Herzberg, Hanan; "Coding for a Channel with Quantization in the Presence of an Estimable Interference," IEEE Transaction On Communications, vol. 45, No. 1, Jan. 1997, pp. 45.51.

* cited by examiner

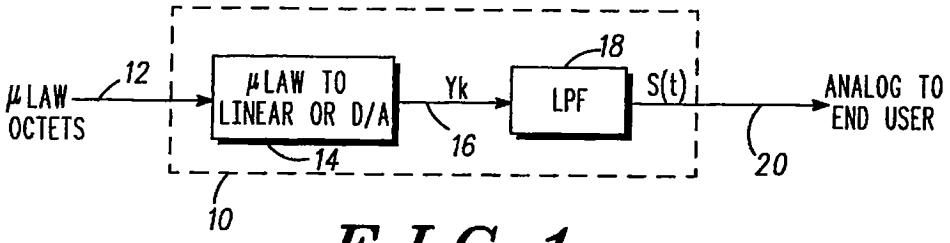


FIG. 1

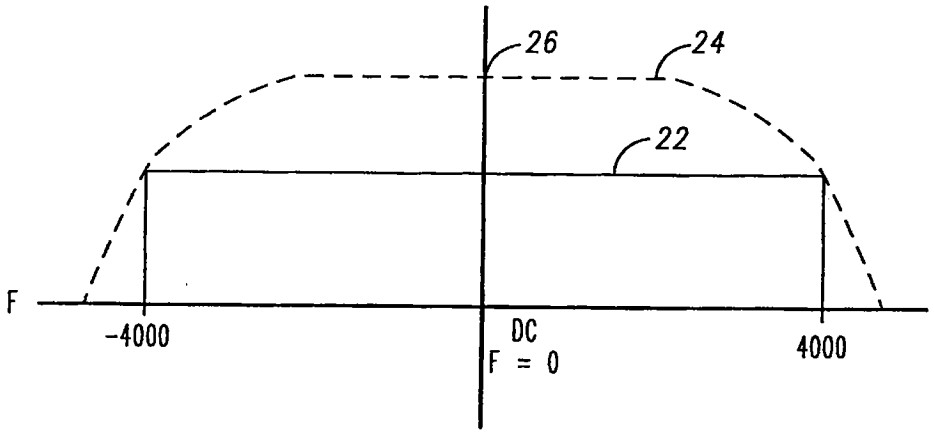


FIG. 2

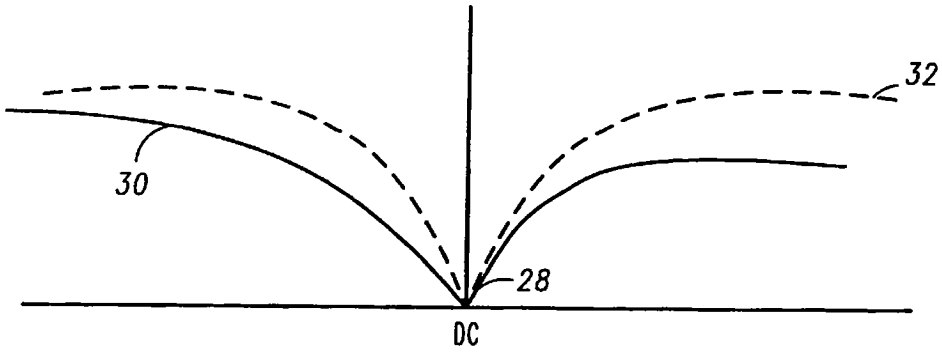


FIG. 3

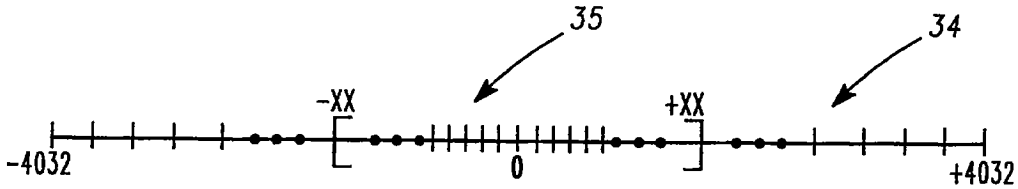


FIG. 4

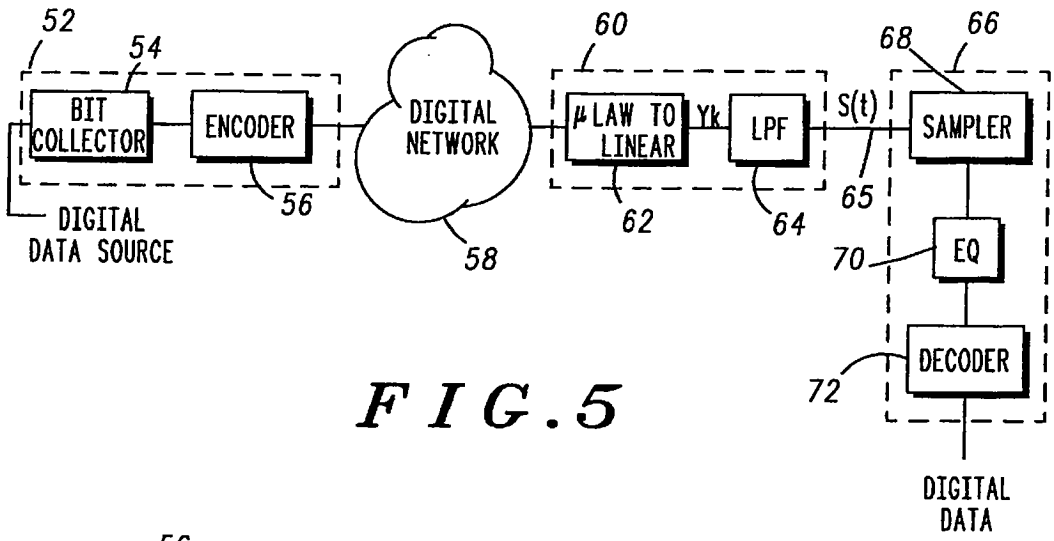


FIG. 5

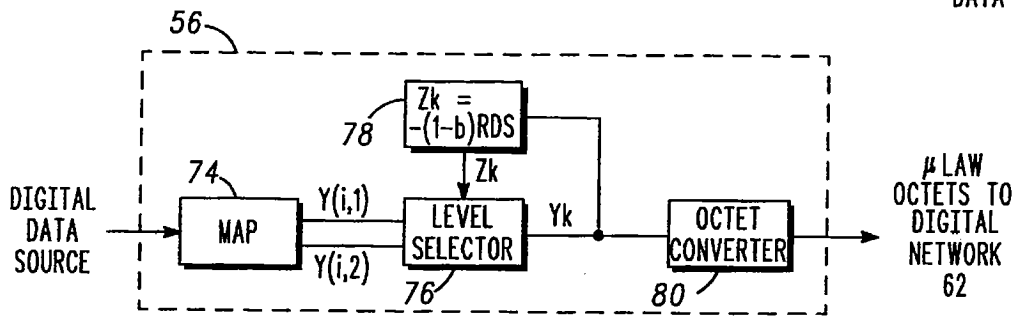


FIG. 6

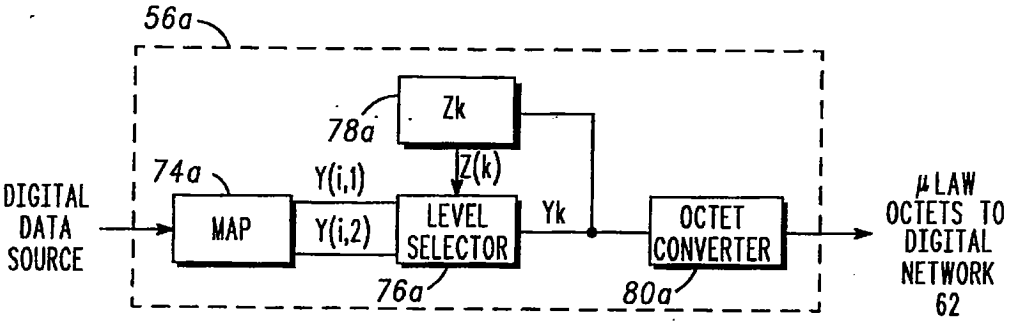


FIG. 7

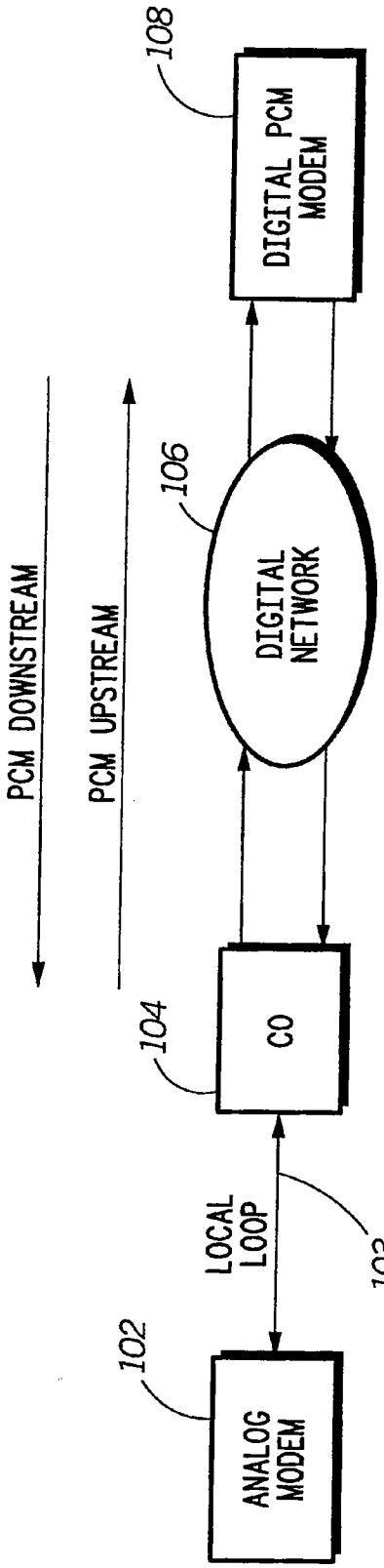


FIG. 8

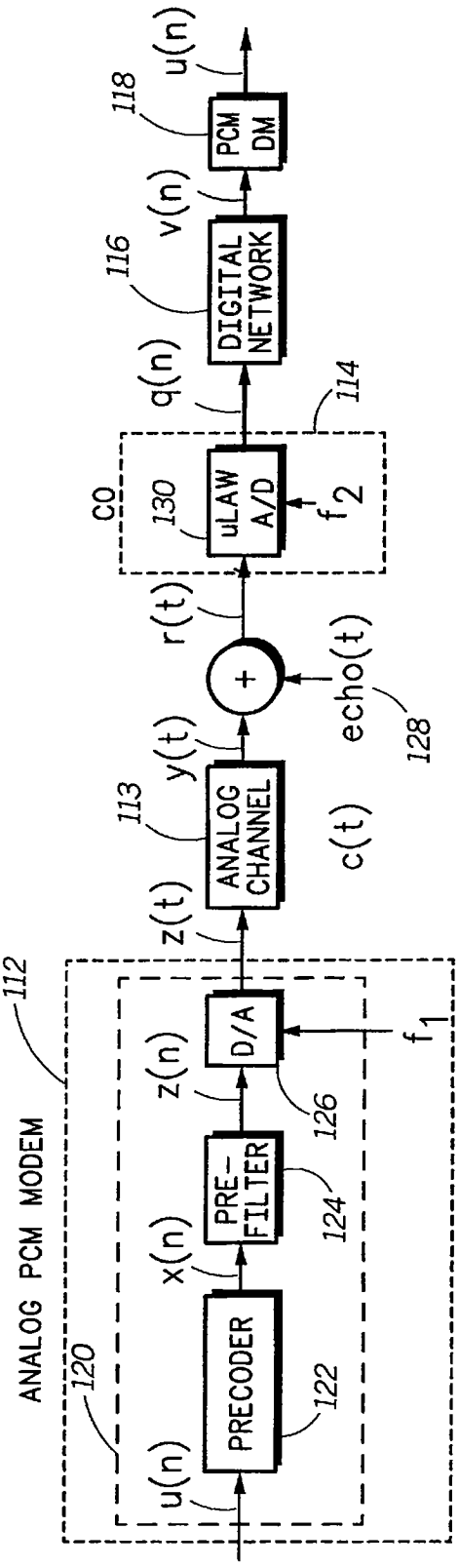


FIG. 9

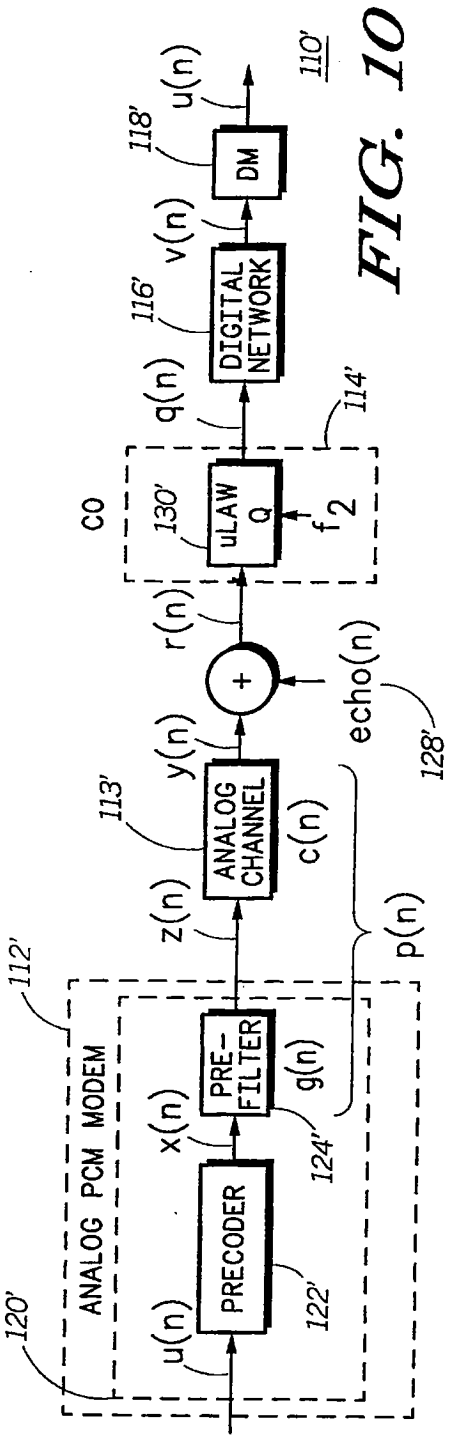


FIG. 10

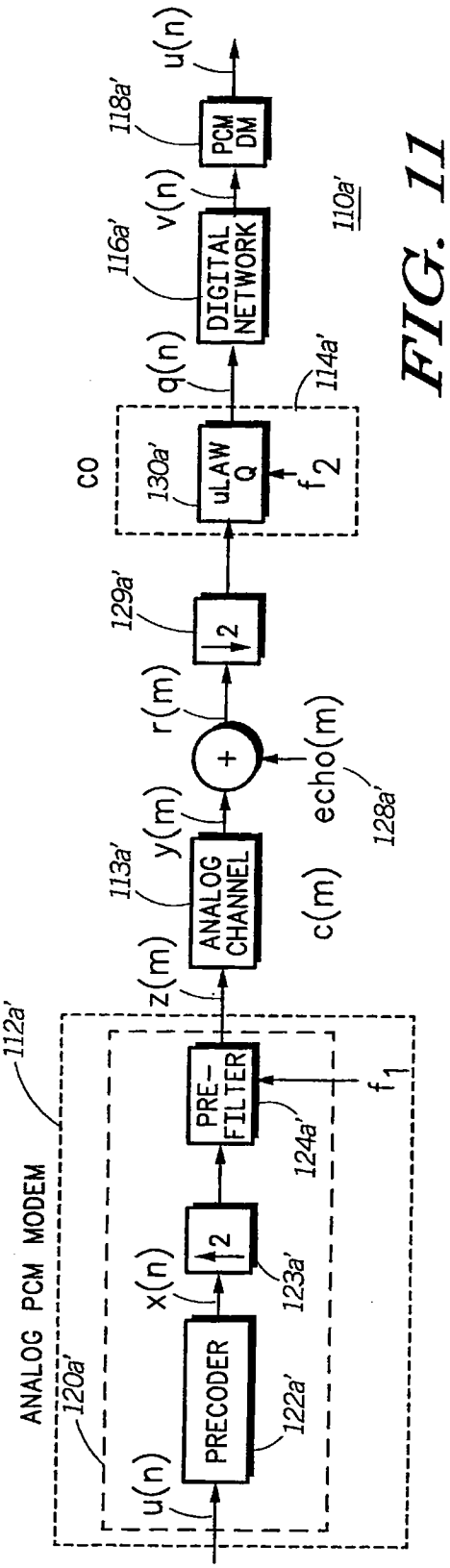


FIG. 11

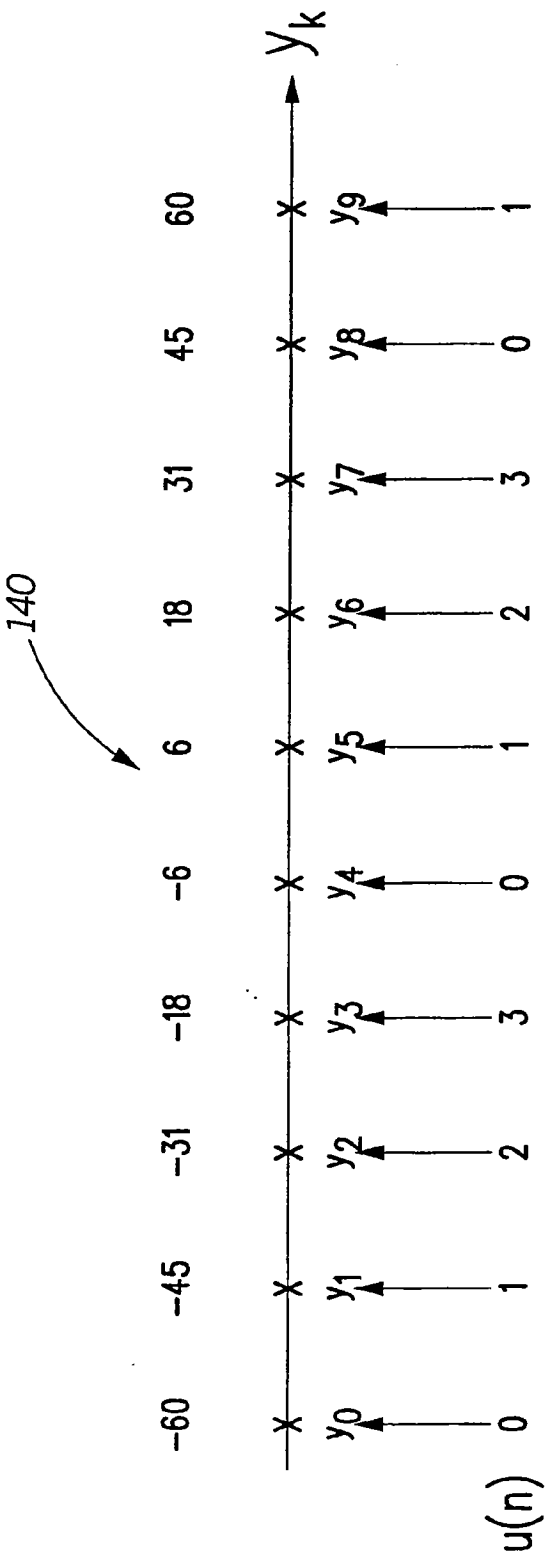


FIG. 12

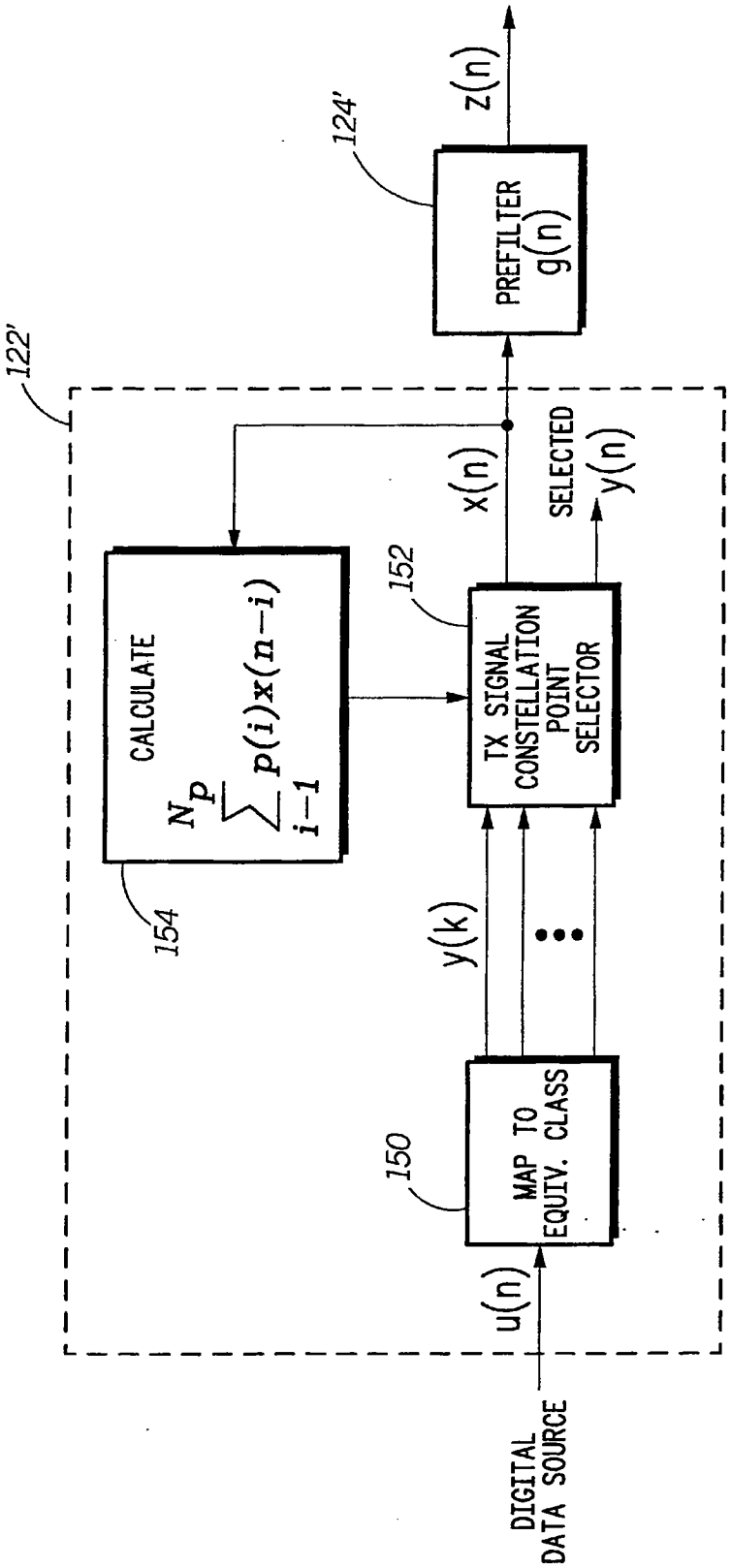


FIG. 13

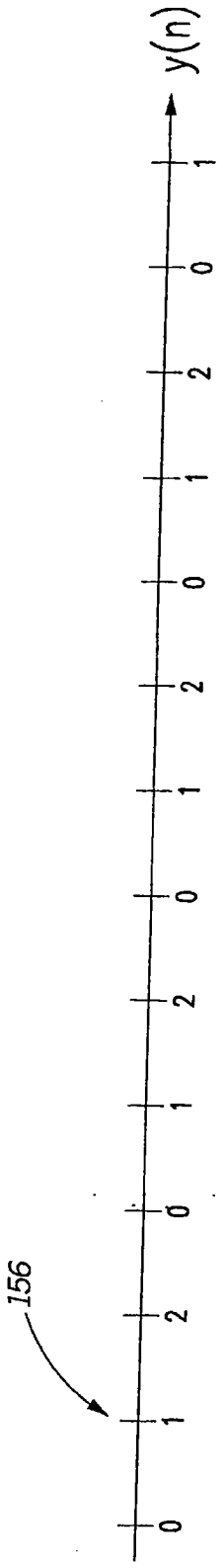


FIG. 14A

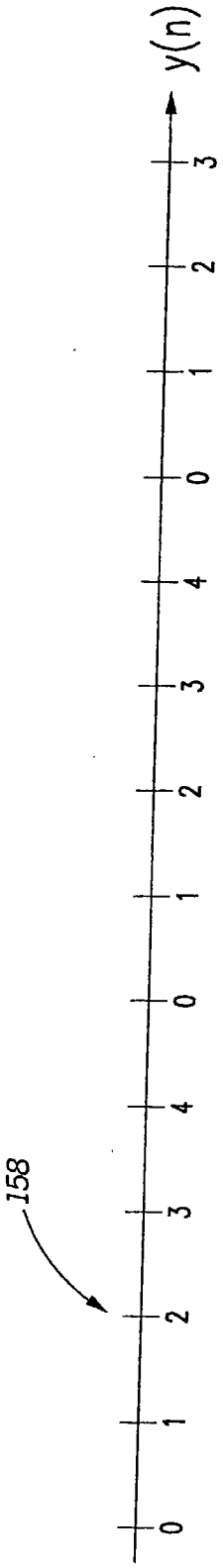
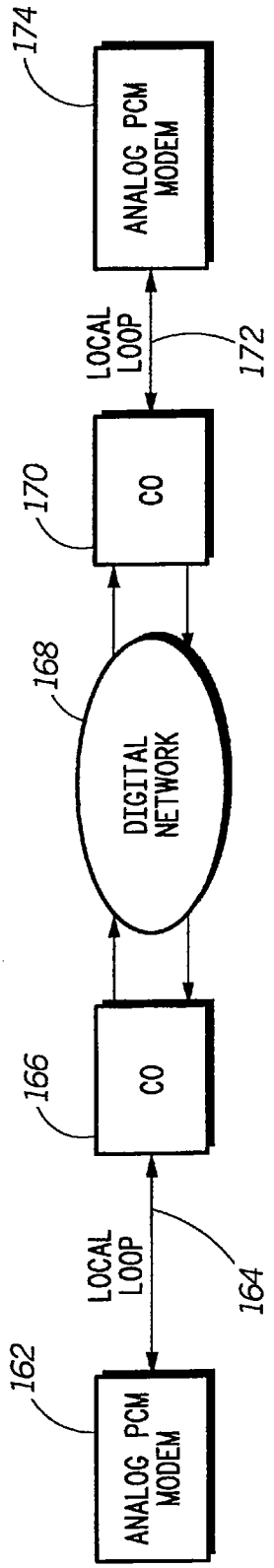


FIG. 14B



PRIOR ART

FIG. 15

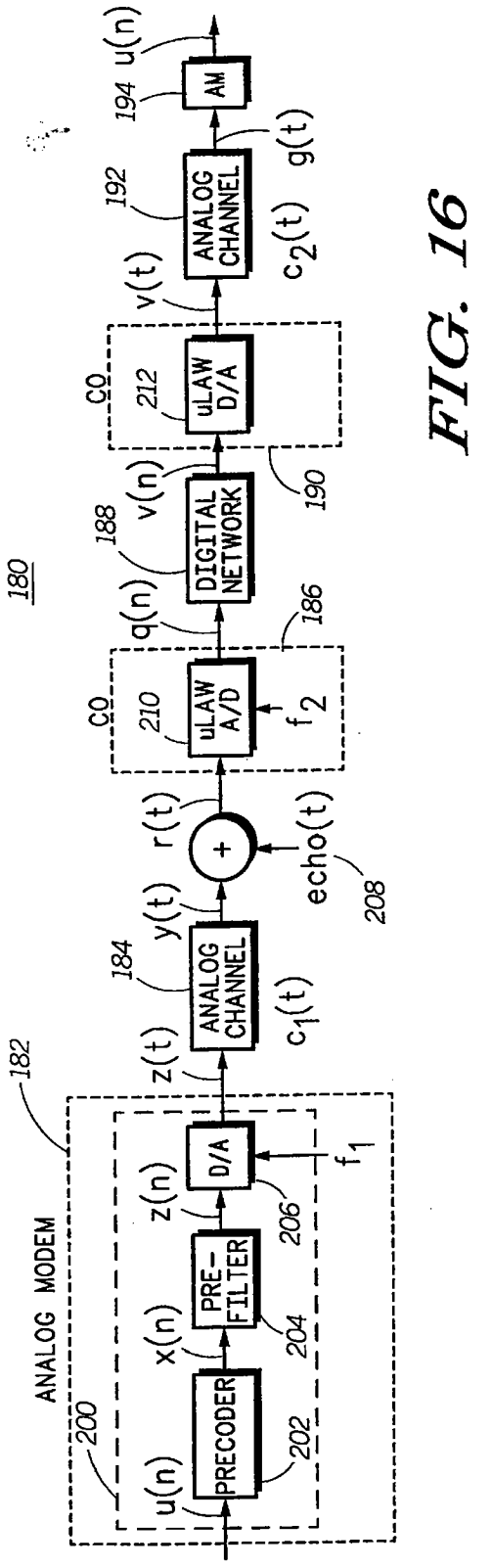


FIG. 16

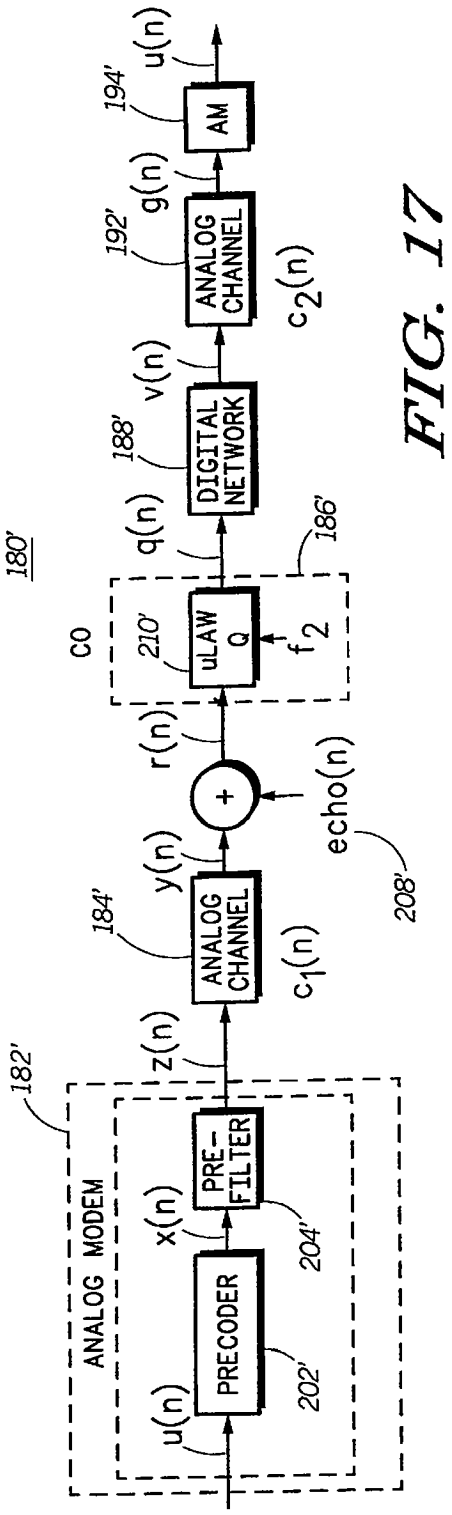


FIG. 17

DEVICE AND METHOD FOR PRECODING DATA SIGNALS FOR PCM TRANSMISSION

RELATED APPLICATIONS

This application is a continuation-in-part of U.S. application Ser. No. 08/747,840, now U.S. Pat. No. 5,818,075 filed Nov. 13, 1996, which is hereby incorporated by reference in its entirety.

FIELD OF INVENTION

This invention relates to a device and method for preceding data signals for pulse code modulation (PCM) transmission.

BACKGROUND OF INVENTION

Conventional modems, such as V.34 modems, treat the public switched telephone network (PSTN) as a pure analog channel even though the signals are digitized throughout most of the network. In contrast, pulse code modulation (PCM) modems take advantage of the fact that most of the network is digital and that typically central site modems, such as those of internet service providers and on-line services, are connected to the PSTN via digital connections (e.g., T1 in the United States and E1 in Europe). First generation PCM modems transmit data in PCM mode downstream only (i.e., from a central site digital modem to an analog end user modem) and transmit in analog mode, e.g. V.34 mode, upstream (i.e., from the end user modem to the central site modem). Future generation PCM modems will also transmit data upstream in PCM mode.

With PCM downstream, the central site PCM modem transmits over a digital network eight bit digital words (octets) corresponding to different central office codec output levels. At the end user's central office, the octets are converted to analog levels which are transmitted over an analog loop. The end user's PCM modem then converts the analog levels, viewed as a pulse code amplitude modulated (PAM) signal, into equalized digital levels. The equalized digital levels are ideally mapped back into the originally transmitted octets and the data the octets represent.

With PCM upstream, the end user PCM modem transmits analog levels over the analog loop corresponding to data to be transmitted. The analog levels are modified by the channel characteristics of the analog loop and the modified levels are quantized to form octets by a codec in the end user's central office. The codec transmits the octets to the PCM central site modem over the digital network. The PCM central site modem determines from the octets the transmitted levels and from the levels the data transmitted by the end user PCM modem is recovered.

A difficulty that exists with upstream PCM transmission is that the levels transmitted by the end user PCM modem are modified by the analog loop. Since these modified levels are the levels that are quantized to form octets by the codec, and not the levels that are actually transmitted, it can be difficult for the central site modem to accurately determine from the octets the data being transmitted by the end user PCM modem. This difficulty is compounded by the fact that there is a channel null in the analog loop, quantization noise introduced by the codec in the end user's central office and downstream PCM echo, which make it more difficult for the central site PCM modem to accurately recover the data transmitted.

Therefore, a need exists for a device and method for precoding data signals for PCM transmission such that the analog levels that are transmitted by the end user PCM modem accurately produce predetermined analog levels (constellation points) at the input to the codec in the end user's central office, which analog levels (constellation points) correspond to the data to be transmitted by the end user PCM modem. Moreover, there is a need for a device, system and method for preceding data signals for PCM transmission which limits the transmit power and combats a channel null introduced by the analog loop and quantization noise introduced by the codec in the end user's central office.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. **1** is a simplified block diagram of a typical telephone company central office;

FIG. **2** is plot of the frequency spectrum of the y_k signals output from the μ-law to linear converter of FIG. **1** and the spectral shape of the low pass filter of FIG. **1**;

FIG. **3** is a plot of a portion of two frequency spectrums each having a null at DC, wherein one spectrum falls off to zero very abruptly at DC and the other spectrum falls off more gradually;

FIG. **4** is a diagrammatic representation of a portion of a typical μ-law constellation;

FIG. **5** is a block diagram of a modem data connection over the telephone system including a transmitter for spectrally shaping signals according to this invention;

FIG. **6** is a block diagram of the encoder of FIG. **6** used specifically for creating a DC null in said analog signals over an analog loop of the PSTN;

FIG. **7** is a block diagram of the encoder of FIG. **6** which may be used generally for modifying, as desired, the frequency spectrum of the signals output from the analog loop to the end user;

FIG. **8** is a block diagram of a typical analog PCM modem to digital PCM modem communication system;

FIG. **9** is a more detailed block diagram depicting PCM upstream transmission according to this invention;

FIG. **10** is an equivalent discrete time block diagram of the block diagram of FIG. **9**;

FIG. **11** is the equivalent discrete time block diagram of the block diagram of FIG. **9** with the analog modem sampling rate twice that of the CO sampling rate;

FIG. **12** is an example of a transmit constellation having equivalence classes according to this invention;

FIG. **13** is a more detailed block diagram of the analog PCM modem transmitter of FIG. **10** according to this invention;

FIG. **14A** is another example of a transmit constellation having equivalence classes according to this invention;

FIG. **14B** is yet another example of a transmit constellation having equivalence classes according to this invention;

FIG. **15** is a block diagram of a typical analog PCM modem to analog PCM modem communication system;

FIG. **16** is a more detailed block diagram depicting PCM transmission with the PCM modem communication system of FIG. **15**; and

FIG. **17** is an equivalent discrete time block diagram of the block diagram of FIG. **16**.

DETAILED DESCRIPTION OF A PREFERRED EMBODIMENT

There is first described below a technique for PCM downstream spectral shaping or preceding of data signals.

Then, there is described a preceding technique for PCM upstream transmission of data signals. Finally, it is described how the PCM upstream preceding technique according to this invention may be generalized for use in a PCM communication system interconnecting two analog PCM modems, as opposed to the typical analog PCM modem and digital PCM modem interconnection.

PCM Downstream Spectral Shaping/Precoding

FIGS. **1** and **2** illustrate the presence of energy near DC in the signals transmitted to a remote user's modem over an analog loop. There is shown in FIG. **1** a portion of a typical telephone central office on a PSTN which receives at input 12 μ-law octets transmitted from a modem (transmitting modem, not shown) directly attached to the digital portion of the telephone system, such as the one described in the co-pending applications referred to above which directly encodes the digital data into octets for transmission. These octets are converted by a D/A converter, also known as a μ-law to linear converter **14**, to a sequence of voltage levels, y_k, each level being one of 255 μ-law levels. The levels are output over line **16** to a LPF **18** which outputs over analog loop towards the remote modem's receiver a filtered analog signal s(t) which is an analog representation of the levels. The analog signal is demodulated and decoded by the receiving modem which outputs a digital bitstream which is an estimate of the originally transmitted data.

The sequence of levels y_k on line **16** from μ-law to linear converter **14** has a flat frequency response **22**, FIG. **2**. The spectral shape **24** of LPF **18** contains a significant amount of energy near DC (f=0) as illustrated at point **26**. Since the sequence y_k has a flat frequency response, the spectrum of the signal s(t) output by filter **18** has the same spectral shape **24** as the filter **18** and therefore the signal s(t) also contains a significant amount of energy near DC. As described above, this energy near DC tends to saturate the transformers on the system which produces unwanted non-linear distortion in the signal s(t) transmitted towards the receiving modem.

In some applications this distortion must be reduced. This can be accomplished by reducing the signal energy near DC in the transmitted signal. Such a DC null **28** is depicted in FIG. **3**. As is known in the state-of-the-art, in order to create this spectral null at DC in the transmitted signal, the running digital sum (RDS) of the transmitted levels y_k (namely, the algebraic sum of all previously transmitted levels) must be kept close to zero. The shape of the spectrum around the DC null **28** can vary from a relatively shallow sloped spectrum **30** to a spectrum **32** which falls off very abruptly at DC. The sharpness of the null depends on how tightly the RDS is controlled.

The present invention accordingly encodes the digital data being transmitted into μ-law octets in a manner that maintains the RDS near zero to create the desired spectral null at DC thereby reducing the non-linear distortion caused by transformer saturation.

To illustrate the method of creating a spectral null, we consider an example of transmitting 6 bits with every symbol y_k. It will be apparent to those skilled in the art that the invention can be used for transmitting any other number of bits per symbol, or when the number of bits per symbol transmitted varies from symbol to symbol. In a system without a spectral null, one first selects a subset of 64 levels from the available 255 μ-law levels such that a minimum distance d_{min} between levels is maintained. These 64 levels are symmetric in the sense that for every positive level there is a negative level of the same magnitude. For example, one can achieve a d_{min} of 32 for an average energy well under −12 dBm**0**, the regulatory limit.

A partial representation of all **255** μ-law levels **34** (128 positive and 127 negative) is shown in FIG. **4**. These levels follow a logarithmic law, with the 64 levels closest to the origin being uniformly spaced between −63 and 63 with a spacing of 2. The next positive and negative segments start at +/−66 and they each contain 16 points spaced by 4. The scale continues with segments of 16 points, each with a spacing of the form 2^n separated from the previous segment by a spacing of $0.75 * 2^n$. The final segments extend between +/−2112 and +/−4032 with a spacing of 128. The set 35 is the set of 64 levels selected from these 255 levels to represent each combination of six bits, i.e. 2^6=64.

In the transmitter, incoming bits are collected in groups of 6, and then mapped into μ-law octets, which represent the desired level. In the central office, the μ-law octets are converted into levels, and the resulting levels are then transmitted. In the receiver, an equalizer compensates for the distortion introduced by the LPF and the local loop, and then a decision device estimates the transmitted level, by selecting the level that is closest to the received point.

In order to achieve spectral shaping in the above example, additional levels are also used, but the minimum distance between levels is still kept at 32. For example, consider the case where 92 levels are used. First, these 92 levels are divided into equivalence classes. There are a number of different ways for generating these equivalence classes. One particularly useful way is described here: we label the levels by integers **0** through **91**, for example by assigning the label **0** to the smallest (most negative) level, the label **1** to the next smallest level, and so on. Then, we define 64 "equivalence classes" by grouping together levels whose labels differ exactly by 64. Such grouping leads to 36 equivalence classes with only one level corresponding to one of 36 innermost levels of smallest magnitude, and 28 equivalence classes with two levels whose labels differ by 64. Other methods for generating the equivalence classes may be used. Each possible combination of 6 bits to be transmitted is then represented by an equivalence class.

For example, the bit combination 000000 may correspond to the first equivalence class which consists of two levels each being represented by a different octet. Note that it is not necessary to use the full dynamic range of the D/A converter. The technique can work with any number of levels, as long as more than 64 levels are used. Of course, the more levels used, the better the desired spectral shape can be achieved. Our experiments indicate that very few additional levels need to be considered for generating a DC null with a relatively sharp notch.

In the above example, since each combination of six information bits is represented by an equivalence class and often there is more than one level in an equivalence class, the information bits must be mapped into one of the levels in a selected equivalence class before an octet representing that level is transmitted. This function is described below with regard to FIGS. **5**–**7**.

Transmitter **52**, FIG. **5**, receives from a digital data source, such as a computer, a bitstream of digital data and with bit collector **54** divides the bits into groups of six, for example. Each six-bit group is provided to encoder **56** which selects the equivalence classes from which the desired levels to achieve the spectral null at DC will be selected. The octets which represent the selected levels are output from encoder **56**, transmitted over digital circuit-switched telephone network **58** and arrive at the remote user's central office **60**. At central office **60**, the octets are converted by μ-law to linear converter **62** to the levels, y_k, which pass through LPF **64** and are output over local analog loop **65** as a signal s(t)

having a spectral null at DC. In receiver **66**, the signal s(t) is sampled by sampler **68**, an equalizer **70** compensates for the distortion introduced by LPF **64** and the local loop, and then a decision device or decoder **72** estimates the transmitted level by selecting the level that is closest to the received point. From the level the decoder **72** determines the equivalence class and then recovers the six information bits by performing an inverse mapping function.

The operation of receiver **66** is essentially unchanged as compared to the receiver described in the co-pending applications referred to above. The only difference is that the receiver now needs to consider a larger set of possible levels and the inverse mapping involves the determination of the equivalence class. Equalizer **70** compensates for the linear distortion introduced by the LPF **64** and the local loop **65**, as described in the co-pending applications. For example, when a linear equalizer is used, the output of the equalizer can be represented as follows:

$$rk=yk+nk \tag{1}$$

where nk is the total noise plus distortion present at the output of the equalizer. Decoder **72** then selects the levels y_k nearest to rk as the decision, determines its equivalence class, and then recovers the six information bits by an inverse map.

If the equalizer includes a maximum-likelihood sequence estimator (e.g., the Viterbi equalizer), then the received signal can be represented in the form

$$rk=\Sigma yk-j\ fj+nk, \tag{2}$$

and this time, the decoder selects the closest sequence $\{y_k\}$ using a Viterbi decoder. For each estimated symbol y_k, the decoder determines its equivalence class and then finds the six information bits via an inverse map.

Encoder **56**, FIG. **6**, includes MAP **74** which is a look-up table containing for each possible combination of the six-bit groups of data received from bit collector **54**, FIG. **5**, levels representing each equivalence class i, where i is an integer between **0** and **63**. Each level, two in this example, y(i,**1**) and y(i,**2**) is provided to level selector **76** where a decision is made as to which level, y_k, is to be transmitted.

This decision is made as follows. First, encoder **56** keeps track of the running digital sum (RDS) of the transmitted levels, y_k, by feeding back the output of level selector **76** to function block **78**. From the previously transmitted levels, y_k, function block **78** calculates the weighted RDS, z_k=-(1-b)RDS, where $0 \leqq b<1$ is weighting factor. Because of D/A nonlinearities, the exact values of the y_k levels may not be known in encoder **56**; however, this should not have a significant effect. It is possible to determine the error and send this information back to encoder **56** to make these calculations more accurate.

Given the group of six bits to be transmitted, level selector **76** selects as the level y_k from the equivalence class {y(i,**1**), y(i,**2**)} the level closest to the weighted RDS. It can be seen that when the RDS is positive, zk will be negative and vice versa. This enables the encoder to choose a level, y_k, from each equivalence class such that when its value is added to the RDS it will bring it closer to zero than the other levels in the equivalence class. After selecting the level yk the octet which represents the level y_k is determined by octet converter **80** and transmitted over the digital network. The value of the transmitted octet can be obtained from a look-up table.

The variable b is a weighting factor that controls the trade-off between the sharpness of the spectral null and the average energy of the transmitted signal. Our analysis has shown that when the number of levels is sufficiently larger than the number of equivalence classes, the sequence y_k will have a spectrum which can be approximated by the filter response h(D)=(1-D)/(1-bD). Clearly, when b=0, we find that h(D)=1-D, which is the well-known Class I Partial Response with a sinusoidal spectral shape having a null at DC. On the other hand, as b approaches 1, the spectrum becomes flat across much of the band except for a very sharp spectral null at DC. It can be seen that for b=0, the average energy of y_k will be twice as large as in the case of a flat spectral shape. As b approaches 1, however, the average energy increase will disappear. In some applications, it may be desirable to keep the constellation expansion, measured by the ratio of the number of levels to the number of equivalence classes. It will be apparent to those skilled in the art that the invention can be used with constellations of any number of levels, and with any smaller number of equivalence classes.

The present invention may be more broadly utilized to spectrally shape, as desired, the analog signals output from the μ-law to linear converter at the central office. The example described above is a specific case of using this invention to reduce the energy of the transmitted signal around DC, but the principals of this invention used in that example can be generalized to spectrally shape signals in numerous ways, for example, to pre-equalize the signals.

A generic version of the encoder of this invention, encoder **56***a*, is shown in FIG. **7**. The only difference between this general case and the special case of a spectral null described above is how the sequence or spectral function zk is generated. Let h(D) be a monic, causal impulse response of a filter representing the desired spectral shape, where D is a delay operator. Suppose we represent the sequences {yk} and {zk} using D-transform notation as y(D) and z(D), respectively. Then, the sequence z(D) can be represented as

$$z(D)=(1-1/h(D))y(D). \tag{3}$$

A close examination of this equation reveals that at a given time k, zk only depends on past values of yk, and therefore can be determined recursively. Thus, for each six bit group, encoder **56***a* determines which level from the associated equivalence class is nearest in value to z_k and selects that level. The octet representing that level is then transmitted. Again, our analysis shows that for sufficiently large number of levels the sequence $\{y_k\}$ transmitted by the central office **60** will have a spectrum closely approximating the spectrum of the filter with response h(D).

The technique described here can also be used in conjunction with a more complex scheme for mapping the information bits to equivalence classes. For example, it can be used in conjunction with shell mapping, a mapping technique used in the V.34 high-speed modem specification.

The examples described above are for an uncoded system. However, the principals can be easily applied to a coded system, for example a trellis coded system. The only difference in this case is that the equivalence classes are further partitioned into subsets, which are used to construct the trellis code.

For example, when a one-dimensional trellis code based on a 4-way set partition is utilized together with the same 64-level signal constellation to send bits per symbol, the equivalence classes are partitioned into subsets as follows: $a_1, b_1, c_1, d_1, a_2, b_2, c_2, d_2, \ldots a_n, b_n, c_n, d_n$. In the example described above, the 64 equivalence classes would be partitioned into four subsets each containing sixteen equivalence classes. The output of a rate-½ convolutional encoder,

e.g. two of the six bits in a group, then determines the subset, and the remaining four "uncoded" bits select the specific equivalence class within the subset. The actual level from the chosen equivalence class in the chosen subset is selected as described above. The operation of the encoder is otherwise unchanged.

Of course, when trellis coding is utilized, the receiver will use a decoder to select the most likely sequence. The trellis decoder may also be an equalizer, jointly decoding the trellis code and equalizing for intersymbol interference.

It may also be possible to use the present invention to enable detection of loss of frame synchronization in a receiver. This can be accomplished by infrequently, but periodically violating the rule for selecting the signal point in a given equivalence class, where the period is chosen to be an integer multiple of the desired framing. A loss of frame synchronization, can be detected in the receiver by monitoring such rule violations. The receiver can also reacquire frame synchronization or may simply request a synchronization pattern (training sequence) from the transmitter.

PCM Upstream Precoding

There is shown in FIG. **8**, a typical PCM communication system **100**. System **100** includes analog PCM modem **102** connected to a telephone company central office (CO) **104** over a local analog loop or channel **103**. There is also included a digital network **106** which is interconnected to CO **104** and to digital PCM modem **108**. With this system, PCM data may be transmitted both in the downstream direction (i.e., from digital PCM modem **108** to analog PCM modem **102**) and in the upstream direction (i.e., from analog PCM modem **102** to digital PCM modem **108**). This type of bidirectional PCM communication system is described in U.S. application Ser. No. 08/724,491, entitled Hybrid Digital/Analog Communication Device, which is assigned to the assignee of the present invention and which is incorporated herein in its entirety by reference.

In the above section a technique for PCM downstream spectral shaping or precoding of data signals is described. In this section there is described a precoding technique for PCM upstream precoding of data signals.

In FIG. **9** there is shown in block diagram **110**, an example of PCM upstream transmission in accordance with this invention. In block diagram **110** there is included analog PCM modem **112** interconnected to analog channel **113**. Analog PCM modem **112** includes transmitter **120** having a precoder **122**, prefilter **124** and a digital to analog converter (D/A) **126**. Precoder **122** receives digital data u(n) and outputs precoded digital data signal x(n). The precoded digital data signal is filtered by prefilter **124** to form signal z(n) which is provided to D/A **126**. D/A **126** converts the filtered signal z(n) to analog form and transmits analog signal, z(t), over analog channel **113**, having a channel characteristic, c(t).

The analog channel modifies the transmitted signal z(t) to form signal y(t). The signal y(t) then encounters downstream PCM echo, echo(t) **128**, that is added to y(t), producing signal r(t). Signal r(t) is received by μ-law (A-law in some countries outside of the US) quantizer **130** in central office (CO) **114** and is quantized according to the μ-law. See International Telecommunications Union, Recommendation G.711, Pulse Code Modulation (PCM) of Voice Frequencies, 1972.

The quantized octets (digital values), q(n), are transmitted over digital network **116** at a frequency of 8 kHz where they may be affected by various digital impairments, as discussed below. The possibly affected octets, v(n), are received by digital PCM modem **118** which ideally decodes the octets, v(n), into their corresponding constellation points, y(t), from which the original digital data, u(n), can be recovered. The decoding of v(n) is described in co-pending application Ser. No. 08/999,254 entitled System, Device and Method for PCM Upstream Transmission Utilizing an Optimized Transmit Constellation, CX097028, which is assigned to the assignee of the present invention and which is incorporated herein in its entirety by reference.

Before data can be transmitted upstream, the clock (f_1) of D/A **126** in analog PCM modem **112** must be synchronized to the clock (f_2) of CO **114**. This can be achieved by learning the clock from the downstream PCM signal (not shown) and synchronizing the clocks using the technique proposed in U.S. Pat. No. 5,199,046, entitled First and Second Digital Rate Converter Synchronization Device and Method, incorporated herein by reference in its entirety. Once the clocks are synchronized, PCM upstream block diagram **110**, FIG. **9**, can be represented as equivalent discrete time block diagram **110'**, FIG. **10**, with like components being represented by the same reference numbers containing a prime ('). In block diagram **110'** it is assumed that $f_1=f_2$; however, it must be noted that f_1 does not have to be equal to f_2 as long as the two clocks are synchronized. When f_1 is equal to f_2,n is the time index for 8 kHz samples, since the clock (f_2) of CO **24** is fixed at that frequency.

An example where f_1 does not equal f_2 is depicted in FIG. **11**. Equivalent discrete time block diagram **110***a'*, FIG. **11**, is the same as equivalent discrete time block diagram **110'**, FIG. **10**, except that there is a 2× up-sampler **123***a'* in transmitter **120***a'* and a 2× down-sampler **129***a'* to account for the fact that $f_1=2f_2$. The variables "m" and "n" are the time indexes for 16 kHz and 8 kHz samples, respectively.

Precoder **122'** and prefilter **124'**, according to this invention, are designed to transmit signal z(n) over analog channel **113** such that predetermined constellation points, y(n), corresponding to digital data u(n) are produced at the input of μ-law quantizer **130'** (in combination with an echo component, echo(n), if present). In other words, the input of μ-law quantizer **130'** is y(n)+e(n) in the presence of echo(n) and just y(n) in the absence of echo(n).

Using the PCM upstream precoding technique described below, or another precoding technique, it is difficult for digital PCM modem **118'** to accurately decode u(n) from v(n) in the presence of echo, quantization and digital impairments without a properly designed transmit constellation of points, y(n). It is described in co-pending application CX097028 how to design the transmit constellation for y(n) to enable y(n) (and eventually u(n)) from v(n)) to be decoded in the presence of echo, quantization and digital impairments with minimized error probability.

As described in co-pending application CX097028, for a given connection, depending on the line conditions, a transmit constellation for each robbed bit signaling (RBS) time slot is selected. As an example, transmit constellation **140** is depicted in FIG. **12**. This constellation includes ten constellation points, y_0–y_9, ranging in value from –39 to 39. It should be noted that the constellation points, y(n), are not necessarily G.711 μ-law levels.

The constellation points y(n) correspond to digital data to be transmitted, u(n). In other words, each constellation point represents a group of data bits and the number of data bits represented by each constellation point depends on the number of points in the constellation (and the number of equivalence classes which are described below). The more points in the constellation, the more bits of data that can be represented. As shown in FIG. **12**, digital data u(n) is divided into four groups of bits **0**,**1**,**2** and **3**, corresponding to 00, 01,

10 and 11, for example. Thus, in this example each constellation point transmitted represents two bits and since the constellation points are transmitted at 8 k/sec, the data rate is 16 kbps. It must be understood that this is a simplified example and data may be mapped into u(n) using any mapping schemes that can map bits into equivalence classes, such as shell mapping or modulus conversion.

According to this invention, the constellation points are grouped into equivalence classes. An equivalence class is a set of typically two or more constellation points which represent the same group of bits or digital data to be transmitted, u(n). With constellation **140**, it is shown that constellation points $y_0(-60)$, $y_4(-6)$, and $y_8(45)$ form the equivalence class for u(n)=0. Constellation points $y_1(-45)$, $y_5(6)$, and $y_9(60)$ form the equivalence class for u(n)=1 and constellation points $y_2(-31)$, and $y_6(18)$ form the equivalence class for u(n)=2. Finally, constellation points $y_3(-18)$, and $y_7(31)$ form the equivalence class for u(n)=3.

Equivalence class selection is generally accomplished as follows. The constellation, with M points, is indexed as y_0, $y_1, \ldots y_{M-1}$ in ascending (or descending) order. Assuming u(n) has U values, e.g. U=4 as in the above example, then the equivalence class for u(n)=u contains all the y_k's where k modulo U is u. For example, in FIG. **11**, the equivalence class for u(n)=0 is y_0, y_u, y_{2U}, where U=4. Note that each equivalence class is not required to have the same number of constellation points.

The number of supporting data levels for u(n) should be chosen to satisfy the following two conditions: 1) The expansion ratio, which is defined as the ratio between the number of constellation points for y(n) and the number of supporting data levels for u(n), i.e., M/U; and2) TX power constraints.

The expansion ratio should be large enough to guarantee stable operation. The size of the expansion ratio will depends on the channel characteristics. In voice band modem applications, there is at least one spectral null at f=0. Therefore, we should have an expansion ratio of $M/U \geqq 2$ to make the system stable. In practice, to guarantee the stability, the quality of the channel is determined from the channel response, c(n), and the minimum expansion ratio is set accordingly. For example, we can use C(f=4 kHz), the frequency response of the channel at 4 kHz (with respect to other frequencies like 2 kHz), as the quality of the channel and depending on that quality we set the minimum expansion ratio. If the C(f=4 kHz) C(f=2 kHz), then we set $M/U \geqq 2.0$. As the C(f=4 kHz) gets smaller and smaller, the expansion ratio must be increased.

As described below, precoder **122'** selects the appropriate constellation point, y_k, from the equivalence class for the data, u(n), to be transmitted and determines a value for x(n) that will produce the selected constellation point at the input to μ-law quantizer **130'**.

The preceding scheme, i.e., the design of precoder **122'** and prefilter **124'**, are now described as follows. From the characteristics of analog channel **113'**, c(n), n=0,1, . . . N_c-1, determined by digital PCM modem **118'**, as described in co-pending application Ser. No. 08/999,416 entitled Device and Method for Detecting PCM Upstream Digital Impairments in a Communication Network, CX097029, which is assigned to the assignee of the present invention and which is incorporated herein in its entirety by reference, an optimal target response p(n), n=0, 1, . . . N_p-1, and corresponding prefilter g(n), $n=-\Delta,-\Delta+1, \ldots, -\Delta+N_g-1$ (where Δ is the decision delay), as shown in FIG. **10**, are determined. This problem is similar to determining the optimal feedforward and feedback filters for a decision feedback equalizer (DFE). The prefilter corresponds to feedforward filter of DFE and the target response corresponds to feedback filter of DFE. See, N. Al-Dhahir, et al, "Efficient Computation of the Delay Optimized Finite Length MMSE-DFE", IEEE Transactions On Signal Processing, vol. 44, no. May 5, 1996, pp.1288–1292. Preferably, the target response p(n) and the filter g(n) will be determined in the analog modem, but they can be determined in the digital modem and transmitted to the analog modem.

The prefilter g(n), $n=-\Delta,-\Delta+1, \ldots, -\Delta+N_g-1$, and the target response p(n), n=0, 1, . . . , N_p-1, (where p(0)=1) can be derived given c(n) by minimizing the cost function ζ as follows:

$$\zeta=\|g(n)*c(n)-p(n)\|^2+\alpha\|g(n)\|^2 \quad (4)$$

The first term ensures small intersymbol Interference (ISI), i.e., the receiver of digital PCM modem **118'** receives what precoder **122'** tried to encode, and the second term enforces the transmit (TX) power to stay finite and small. The term α is a constant term which should be chosen depending on the application. The larger α is the lower the TX power will be, but at the expense of ISI. A smaller α will give less ISI at the expense of TX power. Therefore α should be chosen depending on what is desired for ISI and TX power for a given application. As an example, α can be chosen to be the signal to noise ratio (SNR) of the system, which is $\sigma_n^2/E(x^2)$ or SNR normalized by channel energy, i.e., $SNR/\|c\|^2$. For $E(x^2)$, we can use –9 dBm which is the power constraint for upstream transmission. This minimization problem is the same as DFE tab initialization problem. The term σ_n^2 can be determined as described in co-pending application CX097028.

The initially determined p(n) and g(n) can always be used if the analog channel c(n) is time invariant. However, in practice, c(n) is time variant, though it is very slowly changing. Therefore, some kind of adaptation scheme is necessary. One way to do it is to monitor performance and retrain if the performance goes bad, i.e., re-estimating c(n) in the digital modem **118'** and sending a new c(n) back to analog modem **112'** to recalculate g(n) and p(n). Another way is to feedback the analog channel error signal, error(n), as described in co-pending application CX097029, from digital modem **118'** to analog modem **112'** through downstream data transmission and use that error signal to adapt p(n) and g(n).

Once the target response p(n) is determined precoder **122'** can be implemented. As explained above, we can send data u(n) by transmitting x(n) such as to produce at the input to quantizer **130'**, FIG. **10**, a constellation point y(n) which is one of the points in the equivalence class of u(n). Which constellation point from the equivalence class of u(n) to use to represent u(n) is usually selected to minimize the TX power of transmitter **120'**. The TX power of transmitter **120'** is the power of z(n) (or some other metric). In practice, since it is hard to minimize the power of z(n), the power of x(n) is minimized instead, which is a close approximation of minimizing z(n).

The following is a known relationship among x(n), y(n) and p(n):

$$y(n)=p(n)*x(n) \quad (5)$$

where "*" represents convolution. That relationship can be expressed as follows:

$$y(n)=p(0)x(n)+p(1)x(n-1)+\ldots p(N_p)x(n-N_p) \quad (6)$$

Since p(0) is designed to equal to 1, then equation (6) can be simplified as follows:

$$x(n) = y(n) - \sum_{i=1}^{N_p} p(i)x(n-i). \quad (7)$$

And, since p(n) and the past values of x(n) are known, the appropriate y(n), among the constellation points of the equivalence class of a given u(n), may be selected to minimize $x^2(n)$ in order to minimize the TX power of transmitter **120'**.

Or, lookahead (i.e., decision delay) can be introduced to choose y(n). That is, y(n−Δ) can be chosen from the set of equivalence classes for u(n−Δ) to minimize

$$|x(n-\Delta)|^2+|x(n-\Delta+1)|^2+\ldots+|x(n)|^2,$$

where:

$$x(n-j) = y(n-j) - \sum_{i=1}^{N_p} p(i)x(n-j-i) \quad (8)$$

where j=0, 1, . . . Δ and where y(n−j) is chosen from the set of equivalence classes of u(n−j) (j=0, 1, . . . Δ−1).

Precoder **122'** may be implemented according to this invention as depicted in FIG. **13**. Precoder **122'** includes a mapping device **150** which receives the incoming digital data u(n) from a digital data source and, depending on the number of bits that can be transmitted with each constellation point, determines for each group of bits the equivalence class associated with the group of bits. Mapping device **150** outputs the constellation points, y_k, forming the equivalence class to TX signal/constellation point selector **152** which selects the constellation point, y_k, from the equivalence class and determines the transmit signal x(n) based on the input from calculation device **154**.

Filter device **154** receives the transmit signal x(n) and calculates the summation term (or running filter sum (RFS)) of equation (7) above. Based on the value of the RFS, TX signal/constellation point selector **152** selects the constellation point in the equivalence class that will cause x(n) in equation (7) to be closest in value to zero and calculates the value of x(n) from the calculated RFS and the selected constellation point. The calculated transmit signal x(n) is then provided to prefilter **124'** where x(n) is filtered to form signal z(n) which is transmitted over analog channel **113'**, FIG. **10**.

In order to limit the TX power of transmitter **120'**, FIG. **10**, to keep it within the FCC regulations, the equivalence classes for u(n) must be designed accordingly. With a constellation having a predetermined number of constellation points, If we want to send more data, then more groups of data, u(n), and hence equivalence classes for u(n) will be required. As a result, the constellation points will be further away and will require more transmit power. This is because y(n) is chosen as described below according to equation (7) to minimize $x^2(n)$. Therefore, if the constellation points in the equivalence classes are spaced further apart, it is more likely that $x^2(n)$ will be larger. Thus, to reduce the TX power, we can make the equivalence class of u(n) closer at the expense of rate. This is depicted in FIGS. **14A** and **14B**.

In FIGS. **14A** and **14B**, both constellations **156**, FIG. **14A**, and **158**, FIG. **14B**, have the same number of constellation points; however, constellation **156** has only three equivalence classes u(n)=0,1 and 2 while constellation **158** has five equivalence classes u(n)=0,1,2,3 and 4. Using constellation **158** will require more TX power than constellation **156**, but it will be capable of transmitting at a higher data rate.

The approximate TX power (the power of z(n)) can be calculated as follows when U is the number of points desired to support u(n):

$$P_z \approx |g(n)|^2 \frac{1}{U} \sum_{i=0}^{U-1} dist^2(u(n)=i)/12 \quad (9)$$

where $|g(n)|^2$ is the energy of prefilter and dist(u(n)=i) is the minimum distance between the points in the equivalence points. For example, in FIG. **12** dist(u(n)=0)=|−6−(−60)|=54. Several values of U should be tried to find out the one which satisfies the power constraints. Note also that this should be done for each time slot.

The transmit constellation selection and equivalence class selection according to this invention may be summarized as follows:

1) Obtain digital impairments, calculate noise variance, σ_n^2, and echo variance, σ_e^2, as described in co-pending application CX097028;
2) From σ_e^2, σ_n^2, and the digital impairments, choose the proper constellation for y(n) for each time slot, also as described in co-pending application CX097028; and
3) For each time slot, find the number of points that can be supported for u(n) while satisfying the TX power constraints and the minimum expansion ratio to guarantee stable operation. From this U the constellation for y(n), and the equivalence classes for u(n) can be determined.

The above preceding technique which utilizes a one dimensional constellation can be expanded to multi-dimensional constellations by expanding the definition of the equivalence class of u(n). The following references describe various downstream preceding techniques using multi-dimensional constellations: Eyuboglu, Vedat; "Generalized Spectral Shaping for PCM Modems," Telecommunications Industry Association, TR30.1 Meeting, Norcross, Ga., Apr. 9–11, 1997, pages 1–5; Eyuboglu, Vedat; "Convolutional Spectral Shaping," Telecommunications Industry Association, Tr30.1 Meeting, Norcross, Ga., Apr. 9–11, 1997; Eyuboglu, Vedat; "More on Convolutional Spectral Shaping," ITU Telecommunications Standardization Sector **009**, V.pcm Rapporteur Meeting, La Jolla, Calif., May 5–7, 1997; Eyuboglu, Vedat; "Draft Text for Convolutional Spectral Shaping," ITU-T SG 16 Q23 Rapporteur's Meeting, Sep. 2–11, 1997, Sun River, Oreg.; Eyuboglu, Vedat; "A Comparison of CSS and Maximum Inversion," Telecommunications Industry Association, Tr30.1 Meeting on PCM Modems, Galveston, Tex., Oct. 14–16, 1997; and Eyuboglu, Vedat; "Draft Text for Convolutional Spectral Shaping," Telecommunications Industry Association, Tr30.1 Meeting Galveston, Tex., Oct. 14–16, 1997.

Moreover, the example described above is for an uncoded system. However, the principals can be easily applied to a coded system, for example a trellis coded system. The only difference in this case is that the equivalence classes are further partitioned into subsets, which are used to construct the trellis code.

Generalized PCM Precoding

The above described PCM upstream precoding technique (i.e. from analog PCM modem **112'**, FIG. **10**, to digital PCM modem **118**, may be applied to an analog PCM modem to analog PCM modem connection as depicted in FIG. **15**. System **160** includes analog PCM modem **162** connected to CO **166** over analog loop or channel **164**. CO **166** is interconnected to digital network **168**. Similarly analog PCM modem **174** is connected to CO **170** over analog loop or channel **172**. And, CO **170** is connected to digital network **168**.

Block diagram **180**, FIG. **16**, depicts an analog PCM modem to analog PCM modem connection according to this invention. In block diagram **180** there is included analog PCM modem **182** interconnected to analog channel **184**. Analog PCM modem **182** includes transmitter **200** having a precoder **202**, prefilter **204** and a digital to analog converter (D/A) **206**. Precoder **202** receives digital data u(n) and outputs precoded digital data x(n). The precoded digital data is filtered by prefilter **204** to form signal z(n) which is provided to D/A **206**. D/A **206** converts the filtered signal z(n) to analog form and transmits analog signal, z(t), over analog channel **184**, having a channel characteristic, c(t).

The analog channel modifies the transmitted signal z(t) to form signal y(t). The signal y(t) then encounters PCM echo, echo(t) **208**, that is added to y(t), producing signal r(t). Signal r(t) is received by μ-law (A-law in some countries outside of the US) quantizer **210** in central office (CO) **186** and is quantized according to the μ-law. See International Telecommunications Union, Recommendation G.711, Pulse Code Modulation (PCM) of Voice Frequencies, 1972.

The quantized octets (digital values), q(n), are transmitted over digital network **188** at a frequency of 8 kHz where they may be affected by various digital impairments, as discussed below. The possibly affected octets, v(n), are received by CO **190** and the octets, v(n), are converted by μ-law D/A **212** into analog levels for transmission over analog channel **192**. The levels are received by analog PCM modem **194** which converts the levels to data u(n).

Once the clocks f1 to f2 of D/A **206** and D/A **210** are synchronized, block diagram **180** can be modeled as discrete time block diagram **180'**, FIG. **17**. Analog PCM modem should do the equalization to get v(n) from g(n) in the same way as a downstream PCM modem works as is known in the art. Then, from v(n), a PCM upstream decoding algorithm to decode y(n), i.e. u(n), is undertaken.

The above only describes transmission from analog PCM modem **182'** to analog PCM **194'**; however, transmission in the other direction is accomplished in the same manner. The above described PCM upstream preceding technique (i.e. from analog PCM modem **112'**, FIG. **10**, to digital PCM modem **118**,) can be applied directly to an analog PCM modem to analog PCM modem connection as depicted in FIGS. **15–17**.

It should be noted that this invention may be embodied in software and/or firmware which may be stored on a computer useable medium, such as a computer disk or memory chip. The invention may also take the form of a computer data signal embodied in a carrier wave, such as when the invention is embodied in software/firmware which is electrically transmitted, for example, over the Internet.

The present invention may be embodied in other specific forms without departing from the spirit or essential characteristics. The described embodiments are to be considered in all respects only as illustrative and not restrictive. The scope of the invention is, therefore, indicated by the appended claims rather than by the foregoing description. All changes which come within the meaning and range within the equivalency of the claims are to be embraced within their scope.

What is claimed is:

1. A transmitter which defines an equivalence class at the input of a quantization device for precoding a sequence of analog levels to be transmitted over an analog channel to said quantization device, the precoded sequence forming the input to the quantization device, comprising:

a precoder including a mapping device for mapping data bits to be transmitted to as a sequence of equivalent classes, wherein each equivalence class contains one or more constellation points; and a constellation point selector interconnected to the mapping device which selects a constellation point in each equivalence class to represent the data bits to be transmitted and which transmits a level that produces the selected constellation point to an input of the quantization device.

2. The transmitter of claim **1** further including a filter device, operably coupled to the constellation point selector, which receives at its input previously transmitted levels and provides its output to the constellation point selector.

3. The transmitter of claim **2** wherein the constellation point selector selects the constellation point from each equivalence class based on the output of the filter device.

4. The transmitter of claim **3** further including a prefilter, having a predefined filter response, g(n), for filtering the level transmitted by the constellation point selector.

5. The transmitter of claim **4** wherein the response of the filter device is:

$$\sum_{i=1}^{N_p} p(i)x(n-i)$$

where p(i) is a target response and x(n-i) represents the previously transmitted levels.

6. The transmitter of claim **5** wherein the target response, p(n), and the prefilter response, g(n), are derived from the predetermined response, c(n), of the analog channel.

7. The transmitter of claim **5** wherein the constellation point selector transmits the levels, x(n), according to the following function:

$$x(n) = y(n) - \sum_{i=1}^{N_p} p(i)x(n-i)$$

where y(n) are the constellation points.

8. The transmitter of claim **7** wherein the constellation point selector selects the constellation point in each equivalence class which minimizes the transmit power of the transmitter by selecting the constellation point, y(n), which produces the smallest value for x(n).

9. A method for providing a precoded sequence of analog levels over an analog channel to a quantization device, comprising:

mapping data bits to be transmitted to a sequence of equivalence classes, wherein each equivalence class contains one or more constellation points;

selecting a constellation point in each equivalence class to represent the data bits to be transmitted; and,

transmitting a level that produces the selected constellation point to an input of the quantization device.

10. The transmitter of claim **9** wherein the step of selecting a constellation point includes filtering the previously selected constellation points with a filter device and selecting the constellation points based on the output of the filter device.

11. The method of claim **10** further including filtering the level transmitted with a prefilter having a predefined filter response, g(n).

12. The method of claim 11 wherein the response of the filter device is:

$$\sum_{i=1}^{N_p} p(i)x(n-i)$$

where p(i) is a target response and x(n-i) represents the previously transmitted levels.

13. The method of claim 12 wherein the target response, p(n), and the prefilter response, g(n), are derived from the predetermined response, c(n), of the analog channel.

14. The method of claim 12 wherein step of transmitting includes transmitting the levels, x(n), according to the following function:

$$x(n) = y(n) - \sum_{i=1}^{N_p} p(i)x(n-i)$$

where y(n) are the constellation points.

15. The method of claim 14 wherein the step of selecting includes selecting the constellation point in each equivalence class which minimizes the transmit power of the transmitter by selecting the constellation point, y(n), which produces the smallest value for x(n).

16. A computer useable medium having computer readable program code means embodied therein to function as a precoder for transmitting a precoded sequence of analog levels over an analog channel to a quantization device, comprising:

computer readable program code means for mapping data bits to be transmitted to a sequence of equivalence classes, wherein each equivalence class contains one or more constellation points;

computer readable program code means for selecting a constellation point in each equivalence class to represent the data bits to be transmitted; and,

computer readable program code means for transmitting a level that produces the selected constellation point to an input of the quantization device.

17. The computer useable medium of claim 16 wherein the computer readable program code means for selecting a constellation point includes computer readable program code means for filtering the previously selected constellation points with a filter device and selecting the constellation points based on the output of the filter device.

18. The computer useable medium of claim 17 further including computer readable program code means for filtering the level transmitted with a prefilter having a predefined filter response, g(n).

19. The computer useable medium of claim 18 wherein the response of the filter device is:

$$\sum_{i=1}^{N_p} p(i)x(n-i)$$

where p(i) is a target response and x(n-i) represents the previously transmitted levels.

20. The computer useable medium of claim 19 further including computer readable program code means for deriving the target response, p(n), and the prefilter response, g(n), from the predetermined response, c(n), of the analog channel.

21. The computer useable medium of claim 19 wherein the computer readable program code means for transmitting includes computer readable program code means for transmitting the levels, x(n), according to the following function:

$$x(n) = y(n) - \sum_{i=1}^{N_p} p(i)x(n-i)$$

where y(n) are the constellation points.

22. The computer useable medium of claim 21 wherein the computer readable program code means for selecting includes computer readable program code means for selecting the constellation point in each equivalence class which minimizes the transmit power of the transmitter by selecting the constellation point, y(n), which produces the smallest value for x(n).

23. A computer data signal embodied in a carrier wave, wherein embodied in the computer data signal are computer readable program code means to function as a precoder for transmitting a precoded sequences of analog levels over an analog channel to a quantization device, wherein the analog channel modifies the transmitted analog levels, comprising:

computer readable program code means for mapping data bits to be transmitted to a sequence of equivalence classes, wherein each equivalence class contains one or more constellation points;

computer readable program code means for selecting a constellation point in each equivalence class to represent the data bits to be transmitted; and,

computer readable program code means for transmitting a level that produces the selected constellation point to an input of the quantization device.

24. The computer data signal of claim 23 wherein the computer readable program code means for selecting a constellation point includes computer readable program code means for filtering the previously selected constellation points with a filter device and selecting the constellation points based on the output of the filter device.

25. The computer data signal of claim 24 further including computer readable program code means for filtering the level transmitted with a prefilter having a predefined filter response, g(n).

26. The computer data signal of claim 25 wherein the response of the filter device is:

$$\sum_{i=1}^{N_p} p(i)x(n-i)$$

where p(i) is a target response and x(n-i) represents the previously transmitted levels.

27. The computer data signal of claim 26 further including computer readable program code means for deriving the target response, p(n), and the prefilter response, g(n), from the predetermined response, c(n), of the analog channel.

28. The computer data signal of claim 26 wherein the computer readable program code means for transmitting includes computer readable program code means for transmitting the levels, x(n), according to the following function:

$$x(n) = y(n) - \sum_{i=1}^{N_p} p(i)x(n-i)$$

where y(n) are the constellation points.

29. The computer data signal of claim 28 wherein the computer readable program code means for selecting includes computer readable program code means for select-

ing the constellation point in each equivalence class which minimizes the transmit power of the transmitter by selecting the constellation point, y(n), which produces the smallest value for x(n).

30. In an Analog pulse code modulation (PCM) modem adapted for upstream PCM data transmission to a digital PCM modem, a precoder for preceding a sequence of analog levels transmitted over an analog channel to a quantization device, comprising:

a mapping device for mapping data bits to be transmitted to a sequence of equivalence classes, wherein each equivalence class contains one or more constellation points; and,

a constellation point selector interconnected to the mapping device which selects a constellation point in each equivalence class to represent the data bits to be transmitted and which transmits an analog level that produces the selected constellation point at an input of the quantization device.

31. In an analog pulse code modulation (PCM) modem adapted for PCM data transmission to another analog PCM modem, a transmitter for precoding a sequence of analog levels transmitted over an analog channel to a quantization device, wherein the analog channel modifies the transmitted analog levels, the transmitter comprising:

a mapping device for mapping data bits to be transmitted to a sequence of equivalence classes, wherein each equivalence class contains one or more constellation points; and

a constellation point selector interconnected to the mapping device which selects a constellation point in each equivalence class to represent the data bits to be transmitted and which transmits an analog level that produces the selected constellation point at an input to the quantization device.

* * * * *

UNITED STATES PATENT AND TRADEMARK OFFICE

CERTIFICATE OF CORRECTION

PATENT NO. : 6,198,776 B1
DATED : March 6, 2001
INVENTOR(S) : Eyuboglu, Vedat M. et al.

It is certified that error appears in the above-identified patent and that said Letters Patent is hereby corrected as shown below:

Abstract ,
The 1st line which reads: "A device and method for preceding data signals for pulse.... delete "preceding" and insert -- precoding -- such that the line reads, "A device and method for precoding data signals for pulse"

Signed and Sealed this

Sixteenth Day of October, 2001

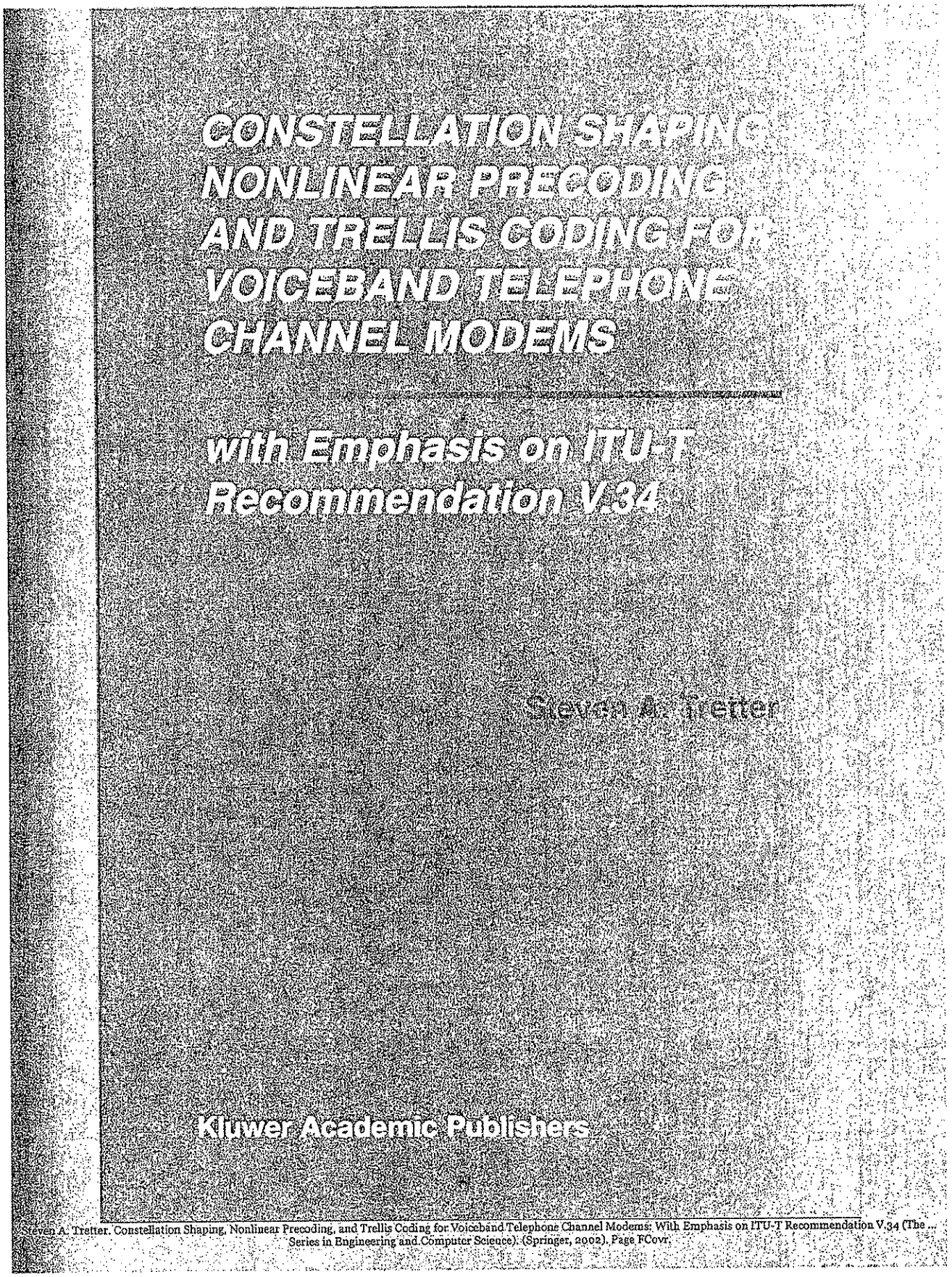
Attest:

Nicholas P. Godici

Attesting Officer

NICHOLAS P. GODICI
Acting Director of the United States Patent and Trademark Office

EXHIBIT E
Part 1



Steven A. Tretter. Constellation Shaping, Nonlinear Precoding, and Trellis Coding for Voiceband Telephone Channel Modems: With Emphasis on ITU-T Recommendation V.34 (The ... Series in Engineering and Computer Science). (Springer, 2002). Page FCovr.

CONSTELLATION SHAPING, NONLINEAR PRECODING, AND TRELLIS CODING FOR VOICEBAND TELEPHONE CHANNEL MODEMS

with Emphasis on ITU-T Recommendation V.34

THE KLUWER INTERNATIONAL SERIES
IN ENGINEERING AND COMPUTER SCIENCE

CONSTELLATION SHAPING, NONLINEAR PRECODING, AND TRELLIS CODING FOR VOICEBAND TELEPHONE CHANNEL MODEMS

with Emphasis on ITU-T Recommendation V.34

Steven A. Tretter
University of Maryland, U.S.A.

KLUWER ACADEMIC PUBLISHERS
Boston / Dordrecht / London

Distributors for North, Central and South America:
Kluwer Academic Publishers
101 Philip Drive
Assinippi Park
Norwell, Massachusetts 02061 USA
Telephone (781) 871-6600 / Fax (781) 681-9045
E-Mail: kluwer@wkap.com

Distributors for all other countries:
Kluwer Academic Publishers Group
Post Office Box 322
3300 AH Dordrecht, THE NETHERLANDS
Telephone 31 786 576 000 / Fax 31 786 576 474
E-Mail: services@wkap.nl


Electronic Services < http://www.wkap.nl>

Library of Congress Cataloging-in-Publication Data

Tretter, Steven A.
Constellation shaping, nonlinear precoding, and trellis coding for voiceband telephone Channel modems with emphasis on ITU-T recommendation V.34 / Steven A. Tretter.
p. cm—(The Kluwer international series in engineering and computer science; SECS 673)
Includes bibliographical references and index.
ISBN 1-40207-006-3 (alk. paper)
1. Modems. 2. Lattice theory. 3. Coding theory. 4. Trellis-coded modulation. I. Title. II. Series.

TK7887.8.M63 T74 2002
621.385'1—dc21

2002016134

Printed on acid-free paper.

Printed in the United States of America.

Contents

Steven A. Tretter. Constellation Shaping, Nonlinear Precoding, and Trellis Coding for Voiceband Telephone Channel Modems: With Emphasis on ITU-T Recommendation V.34 (The ... Series in Engineering and Computer Science). (Springer, 2002). Page TOC4.

Chapter 1

BASICS OF LATTICE THEORY

Most modern quadrature amplitude modulation (QAM) modems use signal constellations consisting of a set of points placed on a regular grid. An infinite regular grid is often called a lattice. These modems include ones for the switched telephone network conforming to ITU-T recommendations such as V.22bis, V.32, V.32bis, and V.34; V.17 facsimile modems; microwave and satellite modems; and a variety of single and multi-carrier digital subscriber line (DSL) modems. Many of these modems use trellis coded modulation (TCM) which involves partitioning the total signal constellation into several levels of subsets, where the subsets at each level have successively fewer points but larger distances between points. This coding method was invented by Ungerboeck [62, 63] and he called the multi-level constellation partitioning technique *set partitioning*. Calderbank and Sloane [9] and Forney [20, 21] formalized the set partitioning idea and put it on a firm mathematical foundation by using the language and theory of lattices. They recognized that set partitioning was equivalent to coset decompositions of lattices and lattice partition chains.

Lattices are important in a wide variety of fields. They occur as real world physical objects like crystals and honeycomb and they have been the subject of extensive deep abstract theoretical investigations in mathematics.

1.1 Definition of a Lattice

Roughly speaking, a lattice is a regular array of points in an N-dimensional space. To define a lattice mathematically, let a *basis* for the lattice be the following set of N linearly independent vectors over an M dimensional space:

$$\begin{aligned} \mathbf{x}_1 &= [x_{1,1}, x_{1,2}, \ldots, x_{1,M}] \\ \mathbf{x}_2 &= [x_{2,1}, x_{2,2}, \ldots, x_{2,M}] \end{aligned}$$

$$\vdots$$

$$\mathbf{x}_N = [x_{N,1}, x_{N,2}, \ldots, x_{N,M}]$$

For our purposes, we will always let $N = M$. Let $i_1, i_2, \ldots, i_N$ be integers. Then, the lattice points are any points with the form

$$\mathbf{v} = i_1\mathbf{x}_1 + i_2\mathbf{x}_2 + \ldots + i_N\mathbf{x}_N \tag{1.1}$$

These definitions can also be expressed in matrix form. Let the *generator matrix* for the lattice be the $N \times M$ matrix

$$\mathbf{G} = \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \vdots \\ \mathbf{x}_N \end{bmatrix} = \begin{bmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,M} \\ x_{2,1} & x_{2,2} & \cdots & x_{2,M} \\ & \vdots & & \\ x_{N,1} & x_{N,2} & \cdots & x_{N,M} \end{bmatrix} \tag{1.2}$$

Then any lattice point has the form

$$\mathbf{v} = [i_1, i_2, \ldots, i_N]\,\mathbf{G} \tag{1.3}$$

The sum of any two lattice points is also a lattice point. This can be seen as follows. Let $\mathbf{v}$ be the lattice point defined by (1.1) and let another point be

$$\mathbf{v}' = i'_1\mathbf{x}_1 + i'_2\mathbf{x}_2 + \ldots + i'_N\mathbf{x}_N$$

Then

$$\mathbf{v} + \mathbf{v}' = (i_1 + i'_1)\mathbf{x}_1 + (i_2 + i'_2)\mathbf{x}_2 + \ldots + (i_N + i'_N)\mathbf{x}_N$$

The coefficients of the basis vectors are integers, so the sum is a lattice point.

The set of lattice points form an algebraic structure known as a commutative group.

1.2 Examples of Lattices

Some examples of simple lattices are discussed in this section. These basic lattices can be used to generate more complicated multi-dimensional lattices. See Conway and Sloane [12] for detailed discussions of many additional types of lattices.

EXAMPLE 1.1 The One-Dimensional Integer Lattice $\mathbf{Z}$

This lattice consists of all points on the real line that are integers. Any one-dimensional lattice is simply a scaled version of $\mathbf{Z}$. ■

EXAMPLE 1.2 The Two-Dimensional Integer Lattice $\mathbf{Z}^2$

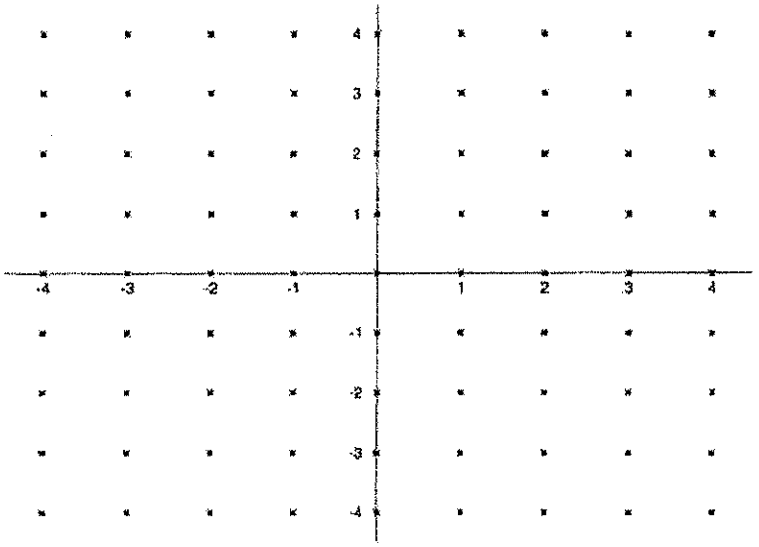


Figure 1.1. A Section of the Two-Dimensional Integer Lattice $\mathbf{Z}^2$

Figure 1.1 shows the 2-dimensional integer lattice $\mathbf{Z}^2$. The lattice consists of all 2-dimensional vectors with integer coordinates. A set of basis vectors for this lattice is

$$\mathbf{x}_1 = [1, 0] \quad \text{and} \quad \mathbf{x}_2 = [0, 1] \tag{1.4}$$

and the corresponding generator matrix is

$$\mathbf{G} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{1.5}$$

The generalization of this lattice is the N-dimensional lattice $\mathbf{Z}^N$ whose points consist of all the N-dimensional vectors with integer coordinates. ■

EXAMPLE 1.3 The Rotated Integer Lattice $R\mathbf{Z}^2$

Figure 1.2 shows the 2-dimensional lattice $R\mathbf{Z}^2$. This lattice can be obtained by rotating $\mathbf{Z}^2$ by 45 degrees and scaling it by $\sqrt{2}$. Thus, R is a rotation operator. Alternatively, it can be generated by starting with $\mathbf{Z}^2$ and deleting all the points

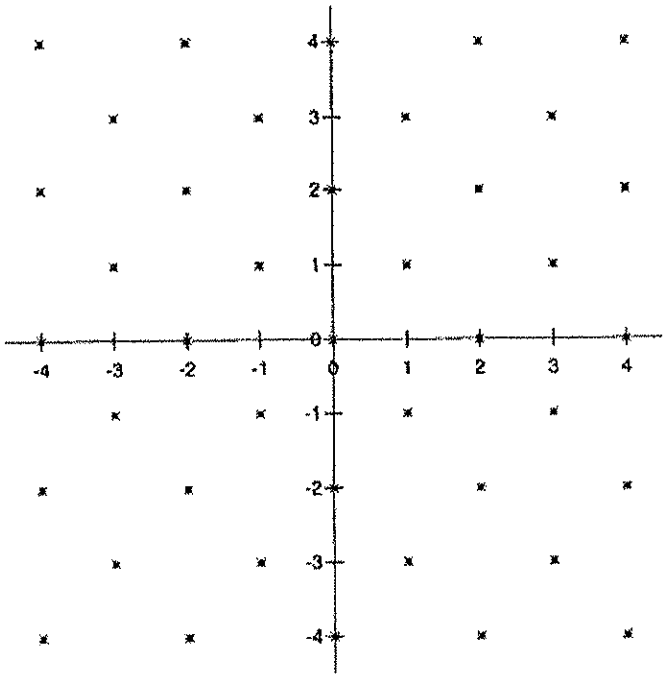


Figure 1.2. A Section of The Lattice $R\mathbf{Z}^2$

along the x-axis with odd x-coordinates, moving up one line and deleting all points with even x-coordinates, etc. Equivalently, the lattice consists of all points with integer coordinates for which the sum of the x and y-coordinates is even. A basis for this lattice is

$$\mathbf{x}_1 = [1, 1] \quad \text{and} \quad \mathbf{x}_2 = [1, -1] \tag{1.6}$$

and the corresponding generator matrix is

$$\mathbf{G} = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \tag{1.7}$$

Let $\mathbf{v}$ be a point in $\mathbf{Z}^2$. Then all points in $R\mathbf{Z}^2$ have the form $\mathbf{vG}$. Clearly, the sum of two points of this form is another point of the same form, so $R\mathbf{Z}^2$ is a lattice. ■

EXAMPLE 1.4 The Scaled Integer Lattice $2\mathbf{Z}^2$

The lattice $2\mathbf{Z}^2$ shown in Figure 1.3 is simply $\mathbf{Z}^2$ scaled by a factor of 2. Notice

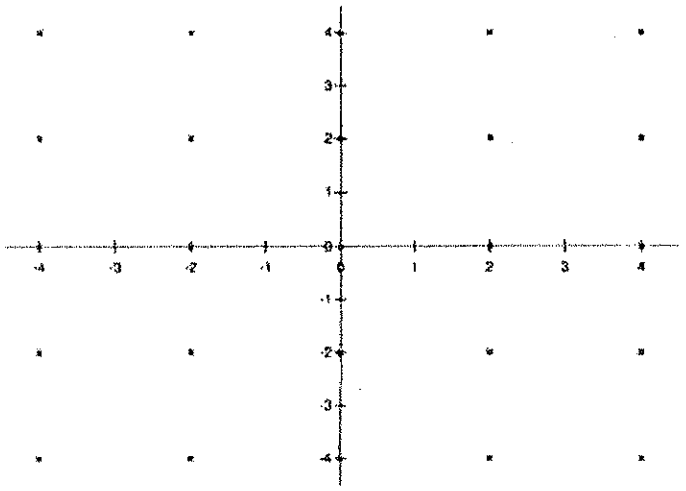


Figure 1.3. A Section of the Lattice $2\mathbf{Z}^2$

that $2\mathbf{Z}^2$ is the same as $R[R\mathbf{Z}^2] = R^2\mathbf{Z}^2$. In general, $c\mathbf{Z}^2$ is $\mathbf{Z}^2$ scaled by a factor of c. A basis for $2\mathbf{Z}^2$ is

$$\mathbf{x}_1 = [2, 0] \quad \text{and} \quad \mathbf{x}_2 = [0, 2] \tag{1.8}$$

■

EXAMPLE 1.5 The 2-Dimensional Hexagonal Lattice $\mathbf{A}_2$

The 2-dimensional hexagonal lattice is shown in Figure 1.4. Notice that each point is surrounded by six closest neighbors that form the vertices of a regular hexagon. This lattice has optimal packing and covering properties in two dimensions [12]. A generator matrix for this lattice is

$$\mathbf{G} = \begin{bmatrix} 1 & 0 \\ 0.5 & 0.5\sqrt{3} \end{bmatrix} \tag{1.9}$$

■

EXAMPLE 1.6 The Four-Dimensional Lattice $\mathbf{D}_4$

The Schläfli lattice, $\mathbf{D}_4$, is the set of integer 4-tuples whose sum of components is even. Equivalently, it can be defined as the set of integer 4-tuples with an even number of odd components. See Conway and Sloane [12] for a detailed

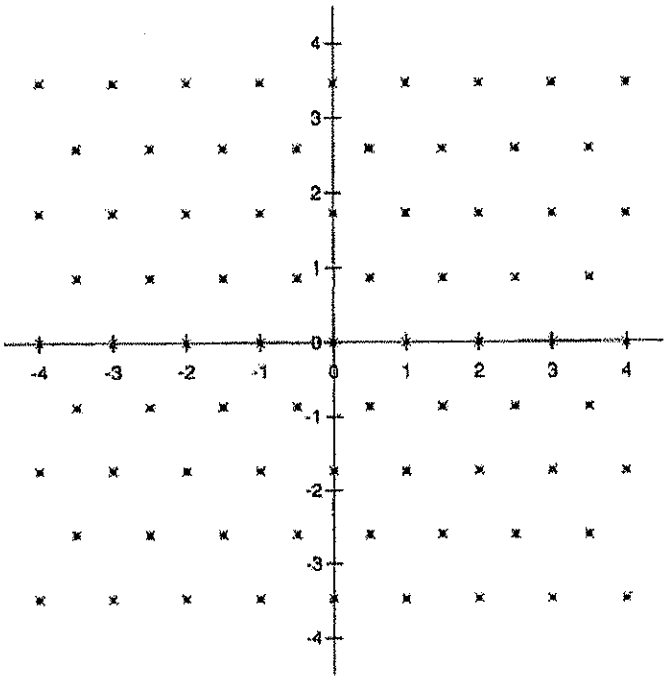


Figure 1.4. A Section of the 2-Dimensional Hexagonal Lattice

discussion of the properties of this lattice. The lattice $\mathbf{D}_4$ is sometimes called the *checkerboard lattice* or the *regular honeycomb*. It generates the densest sphere packing known in four dimensions. One generator matrix for this lattice is

$$\mathbf{G} = \begin{bmatrix} 2 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{bmatrix} \tag{1.10}$$

The generalization of this lattice to N dimensions is the lattice $\mathbf{D}_N$ which consists of the set of integer N-tuples whose sum of components is even. The lattice $R\mathbb{Z}^2$ is also $\mathbf{D}_2$. ■

EXAMPLE 1.7 The Four-Dimensional Lattice $R\mathbb{Z}^4$

In $2N$-dimensions, a $2N$-dimensional "rotation operator" can be defined as the 2-dimensional operator R applied to the N successive pairs of coordinates. For example, when this operator is applied to $\mathbf{Z}^4$, the resulting 4-dimensional lattice

is

$$R\mathbf{Z}^4 = (R\mathbf{Z}^2, R\mathbf{Z}^2) \tag{1.11}$$

Let $\mathbf{v}$ be a point in $\mathbf{Z}^4$ and

$$\mathbf{R}_4 = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 1 & -1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & -1 \end{bmatrix} \tag{1.12}$$

Then points in $R\mathbf{Z}^4$ have the form $\mathbf{v}\mathbf{R}_4$.

Notice that $\mathbf{R}_4^2 = \mathbf{R}_4\mathbf{R}_4 = 2\mathbf{I}$ where $\mathbf{I}$ is the 2×2 identity matrix, so $\mathbf{R}_4^2\mathbf{Z}^4 = 2\mathbf{Z}^4$. The $2N$-dimensional rotation matrix $\mathbf{R}_{2N}$ can be defined similarly and $\mathbf{R}_{2N}^2 = 2\mathbf{I}$ where $\mathbf{I}$ is the $2N$-dimensional identity matrix. Given any $2N$-dimensional lattice Λ, it follows that $\mathbf{R}_{2N}^2\Lambda = 2\Lambda$. ■

1.3 Sublattices, Lattice Partitions, and Cosets

A *sublattice* Λ' of a lattice Λ is a subset of points from Λ that is itself a lattice. That is, the sum of any two points in Λ' is also a point in Λ'. For example, the lattice $R\mathbf{Z}^2$ shown in Figure 1.2 is a sublattice of $\mathbf{Z}^2$. The lattice $2\mathbf{Z}^2$ shown in Figure 1.3 is also a sublattice of $\mathbf{Z}^2$. In addition, $2\mathbf{Z}^2$ is a sublattice of $R\mathbf{Z}^2$.

As stated before, a lattice is an algebraic structure known as a group [4]. A sublattice is an algebraic structure called a *subgroup* of the original group.

Definition. A group consists of the following:

1 A set of elements Λ

2 A rule, which we will call $+$, that associates with each pair of elements λ_1 and λ_2 in Λ an element $\lambda_1 + \lambda_2$ in Λ and has the following properties:

(a) **Associative Law:** For all λ_1, λ_2, and λ_3 in Λ, $\lambda_1 + (\lambda_2 + \lambda_3) = (\lambda_1 + \lambda_2) + \lambda_3$

(b) **Identity Element:** Λ contains an identity element $\mathbf{0}$ such that $\mathbf{0} + \lambda = \lambda + \mathbf{0} = \lambda$ for every λ in Λ.

(c) **Inverses:** For each element λ in Λ there is an inverse element $-\lambda$ in Λ such that $\lambda + (-\lambda) = (-\lambda) + \lambda = \mathbf{0}$.

A translation of a sublattice by an element of the original lattice is called a *coset* of the sublattice. That is, let Λ' be a sublattice of Λ and let λ be an element of Λ. Then the translation $\lambda + \Lambda' = \{\lambda + \lambda' | \lambda' \in \Lambda'\}$ is a coset of Λ' in Λ. The element λ is called a *coset representative*. The coset generated by λ is sometimes designated by $\{\lambda\}$.

Steven A. Tretter. Constellation Shaping, Nonlinear Precoding, and Trellis Coding for Voiceband Telephone Channel Modems: With Emphasis on ITU-T Recommendation V.34 (The ... Series in Engineering and Computer Science). (Springer, 2002). Page 7.

According to the definition of a coset, λ could be chosen as an element of the sublattice Λ'. The resulting coset is then just the sublattice itself. The identity element $\mathbf{0}$ belongs to every lattice, so $\Lambda' = \{\mathbf{0}\}$.

EXAMPLE 1.8

We have observed that the lattice $R\mathbf{Z}^2$ in Figure 1.2 is a sublattice of $\mathbf{Z}^2$. A coset of $R\mathbf{Z}^2$ in $\mathbf{Z}^2$ is the translation $(1,0) + R\mathbf{Z}^2$, which is Figure 1.2 shifted to the right one unit. This is shown in Figure 1.5 where $\Lambda' = R\mathbf{Z}^2$ is shown by circles and the coset by solid squares. The sublattice $\Lambda' = R\mathbf{Z}^2$ is the coset $(0,0) + R\mathbf{Z}^2$. The lattice points $(0,0)$ and $(1,0)$ are one possible pair of coset representatives. The entire set of lattice points in the coset represented by $(0,0)$ is designated by $\{(0,0)\}$ and, similarly, the set of points in the coset represented by $(1,0)$ is represented by $\{(1,0)\}$. Notice that the coset generated by $(1,0)$ is the only coset of $R\mathbf{Z}^2$ distinct from $R\mathbf{Z}^2$ and that the union of $R\mathbf{Z}^2$ and this coset forms $\mathbf{Z}^2$.

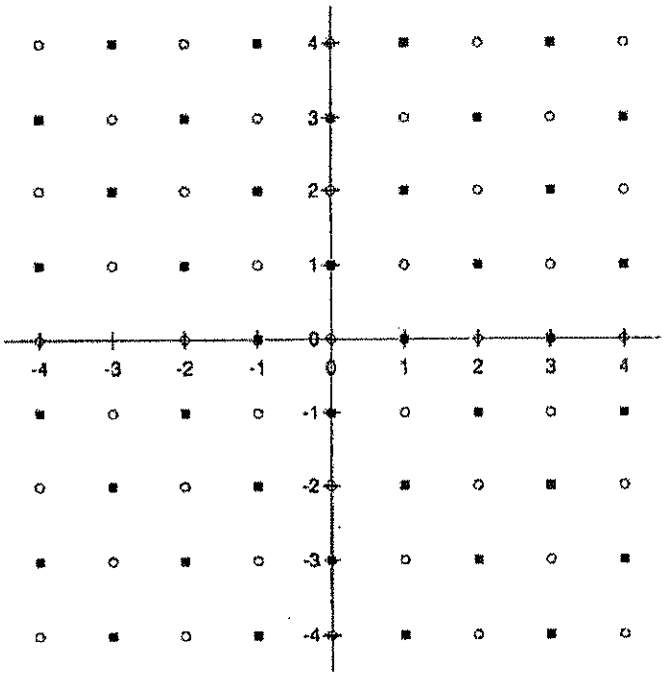


Figure 1.5. Partition of $\mathbf{Z}^2$ into $R\mathbf{Z}^2$ and its Coset

EXAMPLE 1.9

The distinct cosets of the lattice $2\mathbf{Z}^2$ shown in Figure 1.3 with respect to $\mathbf{Z}^2$ are the translations of $2\mathbf{Z}^2$ by the four vectors $(0,0)$, $(1,0)$, $(0,1)$, and $(1,1)$. ∎

EXAMPLE 1.10

The lattice $\mathbf{D}_4$ which consists of all integer 4-tuples whose sum of components is even is a sublattice of $\mathbf{Z}^4$. The set of integer 4-tuples whose sum of components is odd is the only coset of $\mathbf{D}_4$ in $\mathbf{Z}^4$ distinct from $\mathbf{D}_4$. ∎

It can be shown that the union of all the distinct cosets of a sublattice is the original lattice. (In group theory, the union of the cosets of a subgroup is the original group.) The set of distinct cosets of a sublattice Λ' of a lattice Λ is called a *partition* of Λ or, in group theory language, a *quotient group* and is designated by Λ/Λ'. The number of distinct cosets in the partition (including the sublattice) is called the *order* of the partition and is designated by $|\Lambda/\Lambda'|$. Thus the partition of $\mathbf{Z}^2$ generated by $R\mathbf{Z}^2$ is denoted by $\mathbf{Z}^2/R\mathbf{Z}^2$ and has order 2. Also, $\mathbf{Z}^2/2\mathbf{Z}^2$ has order 4.

Let $\{\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_L\}$, where $L = |\Lambda/\Lambda'|$, be a set of distinct coset representatives for the partition Λ/Λ'. This set of coset representatives is sometimes designated by $[\Lambda/\Lambda']$. Therefore

$$\Lambda = \bigcup_{i=1}^{L} \{\mathbf{c}_i + \Lambda'\} = [\Lambda/\Lambda'] + \Lambda' \tag{1.13}$$

The quotient group consisting of the set of cosets generated by partitioning Λ by a sublattice Λ' is actually a group. The elements of this group are the cosets $\{\mathbf{c}_i\}$ for $i = 1, \ldots, |\Lambda/\Lambda'|$. Any element from the coset generated by $\mathbf{c}_i$ has the form $\mathbf{c}_i + \lambda_1$, where $\lambda_1 \in \Lambda'$. Similarly, elements in the coset generated by $\mathbf{c}_j$ have the form $\mathbf{c}_j + \lambda_2$, where λ_2 is also an element of Λ'. Therefore, the sum of an element from coset i and an element from coset j has the form $\mathbf{c}_i + \mathbf{c}_j + \lambda_1 + \lambda_2$. However, both λ_1 and λ_2 are elements of Λ', so $\lambda_1 + \lambda_2 = \lambda'$ is also an element of Λ'. Thus $\mathbf{c}_i + \mathbf{c}_j + \lambda_1 + \lambda_2 = \mathbf{c}_i + \mathbf{c}_j + \lambda'$ which is an element in the coset generated by $\mathbf{c}_k = \mathbf{c}_i + \mathbf{c}_j$. Based on this conclusion, the addition rule, $\boxplus$, for the elements of the quotient group of cosets is

$$\{\mathbf{c}_i\} \boxplus \{\mathbf{c}_j\} = \{\mathbf{c}_i + \mathbf{c}_j\} \tag{1.14}$$

Since vector addition is associative, it follows that $\boxplus$ is also associative. The coset $\{\mathbf{0}\}$ is the identity element. For each coset representative $\mathbf{c}_i$, its negative $-\mathbf{c}_i$ is an element of some coset $\{\mathbf{c}_j\}$, so $\{\mathbf{c}_i\} \boxplus \{\mathbf{c}_j\} = \{\mathbf{c}_i - \mathbf{c}_i\} = \{\mathbf{0}\}$ and

Steven A. Tretter. Constellation Shaping, Nonlinear Precoding, and Trellis Coding for Voiceband Telephone Channel Modems: With Emphasis on ITU-T Recommendation V.34 (The ... Series in Engineering and Computer Science). (Springer, 2002). Page 9.

each element has an inverse. Thus the quotient group satisfies all the abstract group axioms.

The elements of the lattice partition Λ/Λ' can themselves be partitioned with respect to a sublattice Λ'' of Λ'. The sublattice Λ'' generates the partition Λ'/Λ'' of Λ' with $|\Lambda'/\Lambda''|$ elements. In terms of the notation presented above, $[\Lambda'/\Lambda'']$ is a set of coset representative for this partition and $\Lambda' = [\Lambda'/\Lambda''] + \Lambda''$. Consequently, the original lattice Λ can be represented by

$$\Lambda = [\Lambda/\Lambda'] + \Lambda' = [\Lambda/\Lambda'] + [\Lambda'/\Lambda''] + \Lambda'' \tag{1.15}$$

In words, any element of the lattice Λ can be expressed as a sum of a coset representative from the first level partition plus a representative from the second level partition plus an element of the sublattice Λ''. The original lattice and its two sublattices can be represented by the *partition chain* $\Lambda/\Lambda'/\Lambda''$. This chain induces the partition of Λ given by (1.15). The number of elements in the overall partition is

$$|\Lambda/\Lambda''| = |\Lambda/\Lambda'| \times |\Lambda'/\Lambda''| \tag{1.16}$$

EXAMPLE 1.11

The lattice $\Lambda'' = 2\mathbf{Z}^2$ is a sublattice of $\Lambda' = R\mathbf{Z}^2$ and generates a partition of Λ' of order 2. It can be obtained by rotating $R\mathbf{Z}^2$ by 45 degrees and scaling by the square root of 2. The sequence of partitions of $\mathbf{Z}^2$ with respect to $R\mathbf{Z}^2$ and $R\mathbf{Z}^2$ with respect to $2\mathbf{Z}^2$ is induced by the partition chain $\mathbf{Z}^2/R\mathbf{Z}^2/2\mathbf{Z}^2$. Notice that the order of the partition of the first lattice in the chain, $\mathbf{Z}^2$, generated by the last lattice, $2\mathbf{Z}^2$, is the product of the orders of the subpartitions in the chain, that is, $|\mathbf{Z}^2/2\mathbf{Z}^2| = |\mathbf{Z}^2/R\mathbf{Z}^2| \times |R\mathbf{Z}^2/2\mathbf{Z}^2| = 2 \times 2 = 4$. ■

EXAMPLE 1.12

The lattice $R\mathbf{Z}^4$ is a sublattice of $\mathbf{D}_4$. According to (1.11), each point in $R\mathbf{Z}^4$ is the concatenation of a pair of points from $R\mathbf{Z}^2$. The sum of the coordinates of any point in $R\mathbf{Z}^2$ is even. Therefore, the sum the four coordinates of a point in $R\mathbf{Z}^4$ must be even and the point must be an element of $\mathbf{D}_4$.

However, $\mathbf{D}_4$ contains the additional points where the sum of the left two coordinates is odd and the sum of the right two coordinates is also odd. Then the sum of all four coordinates is even since the sum of two odd numbers is even. These points can be represented as

$$(R\mathbf{Z}^2, R\mathbf{Z}^2) + (1,0,1,0) = R\mathbf{Z}^4 + (1,0,1,0) \tag{1.17}$$

The vector $(1,0,1,0)$ is not unique. Notice that $(-1,1,-1,1)$ is a member of $R\mathbf{Z}^4$ so one of many other representations is

$$R\mathbf{Z}^4 + (1,0,1,0) = [R\mathbf{Z}^4 + (-1,1,-1,1)] + (1,0,1,0)$$

$$= R\mathbf{Z}^4 + (0,1,0,1) \tag{1.18}$$

The observations made above in this example prove that

$$\mathbf{D}_4 = R\mathbf{Z}^4 \bigcup \{R\mathbf{Z}^4 + (1,0,1,0)\} \tag{1.19}$$

and so the partition $\mathbf{D}_4/R\mathbf{Z}_2$ has order 2. ∎

More generally, a *partition chain* $\Lambda/\Lambda_1/\Lambda_2/\ldots/\Lambda_N$ is sequence of lattices where each is a sublattice of the one to the left. This chain induces the partition

$$\Lambda = [\Lambda/\Lambda_1] + [\Lambda_1/\Lambda_2] + \cdots + [\Lambda_{N-1}/\Lambda_N] + \Lambda_N \tag{1.20}$$

The number of elements in the overall partition is

$$|\Lambda/\Lambda_N| = |\Lambda/\Lambda_1| \times |\Lambda/\Lambda_2| \times \cdots \times |\Lambda_{N-1}/\Lambda_N| \tag{1.21}$$

EXAMPLE 1.13 A Partition Chain Used in V.34 Trellis Codes

$\mathbf{D}_4$ is a sublattice of $\mathbf{Z}^4$ according to Example 1.10. Also $R\mathbf{Z}^4$ is a sublattice of $\mathbf{D}_4$ according to Example 1.12. Therefore, we have the partition chain $\mathbf{Z}^4/\mathbf{D}_4/R\mathbf{Z}^4$. Applying the rotation operator R to both sides of the partition $\mathbf{Z}^4/\mathbf{D}_4$, we see that $R\mathbf{Z}^4/R\mathbf{D}_4$ is also a partition. Continuing in this way gives the partition chain

$$\mathbf{Z}^4/\mathbf{D}_4/R\mathbf{Z}^4/R\mathbf{D}_4/2\mathbf{Z}^4/2\mathbf{D}_4 \tag{1.22}$$

Each partition in this chain has order 2, so the partition $\mathbf{Z}^4/2\mathbf{D}_4$ has order $2^5 = 32$. This partition chain is used in the V.34 modem trellis codes. ∎

1.4 Binary Lattices and Coset Representatives

The most useful lattices for generating trellis codes have been a type known as a *binary* lattice. An N-dimensional integer lattice Λ is a binary lattice if it has $2^m\mathbf{Z}^N$ as a sublattice for some integer m. The smallest integer m such that $2^m\mathbf{Z}^N$ is a sublattice of Λ is called the *2-depth* of the lattice. For example, $2\mathbf{Z}^2$ is a sublattice of $R\mathbf{Z}^2$, so the 2-depth of $R\mathbf{Z}^2$ is $m = 1$. The binary lattices with $m = 1$ are called *mod-2 lattices*.

Let Λ' be a sublattice of Λ and assume that both are binary N-dimensional lattices of 2-depth m. Then we can form the partition chain

$$\mathbf{Z}^N/\Lambda/\Lambda'/2^m\mathbf{Z}^N$$

With a little thought, you will see that $|\mathbf{Z}^N/2^m\mathbf{Z}^N| = 2^{mN}$. Thus, each subpartition in the chain must have an order which is a power of 2 since the order of the overall partition is the product of the orders of the subpartitions.

Steven A. Tretter. Constellation Shaping, Nonlinear Precoding, and Trellis Coding for Voiceband Telephone Channel Modems: With Emphasis on ITU-T Recommendation V.34 (The ... Series in Engineering and Computer Science). (Springer, 2002). Page 11.

The *redundancy* $r(\Lambda)$ of an N-dimensional binary lattice Λ is

$$r(\Lambda) = \log_2 |\mathbf{Z}^N/\Lambda| \tag{1.23}$$

Therefore, $|\mathbf{Z}^N/\Lambda| = 2^{r(\Lambda)}$. The *normalized redundancy* per two dimensions is defined to be

$$\rho(\Lambda) = \frac{2}{N} r(\Lambda) \tag{1.24}$$

Using (1.52), it follows that the fundamental volume, which is defined in Section 1.5, for Λ is $V(\Lambda) = 2^{r(\Lambda)}$ and the fundamental coding gain, which is defined in Section 1.7, is

$$\gamma_c(\Lambda) = 2^{-\rho(\Lambda)} d^2_{\min}(\Lambda) \tag{1.25}$$

Ungerboeck's [62, 63] technique for generating a trellis code involves partitioning a signal constellation into subsets by a sequence of two-way partitions so that at each level of partitioning, the minimum Euclidean distance in a subset is greater than the minimum distance in subsets at the previous level. At each level, the number of subsets is a power of 2. The selection of the subsets at each level is controlled by a bit in the output of a convolutional encoder. Many of Ungerboeck's examples consist of generating subsets which are the cosets in the partition of a 2-dimensional mod-2 lattice Λ with respect to a sublattice Λ'.

A method for specifying coset representatives for such a partition will now be presented. Suppose that Λ and Λ' are binary lattices and the partition Λ/Λ' has order 2^L. Then the cosets can be represented by a set of 2^L coset representatives which can be labeled by a binary L-tuple $\mathbf{a} = (a_{L-1}, \ldots, a_1, a_0)$. Let the coset representative (which is an N-dimensional vector) be designated by $\mathbf{c}(\mathbf{a})$ with $\mathbf{c}(\mathbf{0}) = \mathbf{0}$. It was shown in Section 1.3 that the cosets form an additive group with the addition rule

$$\{\mathbf{c}(\mathbf{a}_1)\} \boxplus \{\mathbf{c}(\mathbf{a}_2)\} = \{\mathbf{c}(\mathbf{a}_1) + \mathbf{c}(\mathbf{a}_2)\} = \{\mathbf{c}(\mathbf{a}_3)\} \tag{1.26}$$

The *order of a group* is the number of elements in the group. Suppose $\mathbf{G}$ is a group with finite order and $\mathbf{g}$ is an element in $\mathbf{G}$. Let $n\mathbf{g} = \mathbf{g} + \mathbf{g} + \cdots + \mathbf{g}$ where there are n $\mathbf{g}$'s in the sum. This sequence must repeat for some n so that it must become $\mathbf{0}$ just before becoming $\mathbf{g}$ again. The order of $\mathbf{g}$ is defined to be the smallest n such that the sum is $\mathbf{0}$. It can be shown that in every group with a finite number of elements, there is at least one element whose order is the number of elements in the group and these elements are called *primitive* elements. Furthermore, it can be shown that for every factor of the order of the group, there is an element with that order.

We concluded for the binary lattices Λ and Λ' described above that the partition Λ/Λ' consists of a number of cosets that is a power of 2. Thus, the quotient group must contain an element with order 2 and this element is not

Steven A. Tretter. Constellation Shaping, Nonlinear Precoding, and Trellis Coding for Voiceband Telephone Channel Modems: With Emphasis on ITU-T Recommendation V.34 (The ... Series in Engineering and Computer Science). (Springer, 2002). Page 12.

$\{\mathbf{0}\}$. Equivalently, there must be a coset representative, say, $\mathbf{g}_{L-1}$, which is in Λ but not in $\Lambda' = \{\mathbf{0}\}$ and has the property

$$\{\mathbf{g}_{L-1} + \mathbf{g}_{L-1}\} = \{\mathbf{0}\}$$

Let the lattice Λ_{L-1} be the union of Λ' and its coset $\Lambda' + \mathbf{g}_{L-1}$, that is,

$$\Lambda_{L-1} = \bigcup_{a_{L-1} \in \{0,1\}} \{\Lambda' + a_{L-1}\mathbf{g}_{L-1}\} \tag{1.27}$$

To see that Λ_{L-1} is a lattice, first consider two points λ_1 and λ_2 in Λ'. They are also in Λ_{L-1}, so $\lambda_1 + \lambda_2 \in \Lambda' \subset \Lambda_{L-1}$. Next let λ_1 be an element of Λ' and λ_2 an element of $\Lambda' + \mathbf{g}_{L-1}$. Then $\lambda_1 + \lambda_2 \in \lambda_1 + \Lambda' + \mathbf{g}_{L-1} = \Lambda' + \mathbf{g}_{L-1} \subset \Lambda_{L-1}$. Finally, let λ_1 and λ_2 both be elements of $\Lambda' + \mathbf{g}_{L-1}$. Then for some λ_1' and λ_2' in Λ', $\lambda_1 + \lambda_2 = (\lambda_1' + \mathbf{g}_{L-1}) + (\lambda_2' + \mathbf{g}_{L-1}) \in \Lambda'$ since $\lambda_1' + \lambda_2' \in \Lambda'$ and $\mathbf{g}_{L-1} + \mathbf{g}_{L-1} \in \{\mathbf{0}\} = \Lambda'$.

Repeating the above argument, there must be a coset representative $\mathbf{g}_{L-2}$ of order 2 in Λ but not in Λ_{L-1}, so that a lattice Λ_{L-2} can be formed as

$$\begin{aligned} \Lambda_{L-2} &= \bigcup_{a_{L-2} \in \{0,1\}} \{\Lambda_{L-1} + a_{L-2}\mathbf{g}_{L-2}\} \\ &= \bigcup_{a_{L-1}, a_{L-2} \in \{0,1\}} \{\Lambda' + a_{L-2}\mathbf{g}_{L-2} + a_{L-1}\mathbf{g}_{L-1}\} \end{aligned} \tag{1.28}$$

Continuing this process, it follows that the original lattice can be represented as the following union of 2^L cosets:

$$\begin{aligned} \Lambda &= \bigcup_{\mathbf{a} \in \{0,1\}^L} \{\Lambda' + a_0\mathbf{g}_0 + a_1\mathbf{g}_1 + \cdots + a_{L-1}\mathbf{g}_{L-1}\} \\ &= \bigcup_{\mathbf{a} \in \{0,1\}^L} \{\Lambda' + \mathbf{c}(\mathbf{a})\} \end{aligned} \tag{1.29}$$

where $\mathbf{a} = [a_{L-1}, a_{L-2}, \ldots, a_0]$. The $\mathbf{g}$'s are called *generators* for the partition.

This rule for assigning labels can be compactly expressed in matrix form. Let an $L \times N$ generator matrix $\mathbf{G}$ be formed by stacking the generators (which are considered to be row vectors) vertically as follows

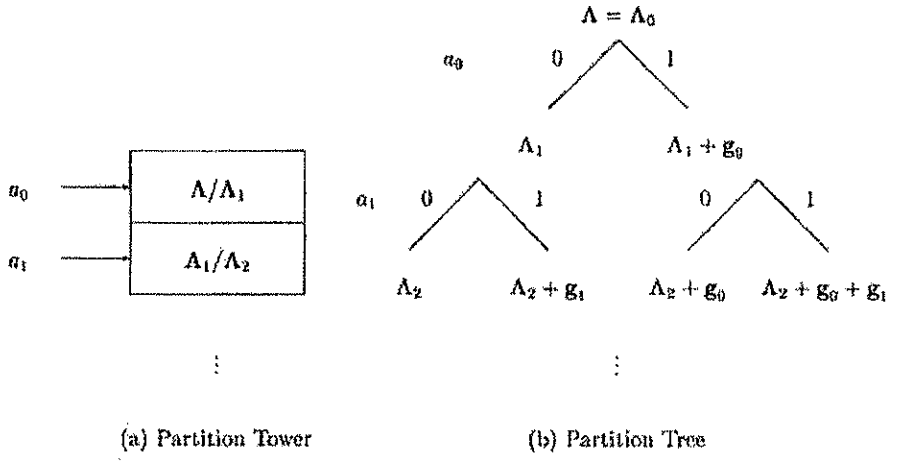
$$\mathbf{G} = \begin{bmatrix} \mathbf{g}_{L-1} \\ \mathbf{g}_{L-2} \\ \vdots \\ \mathbf{g}_0 \end{bmatrix} \tag{1.30}$$

Steven A. Tretter. Constellation Shaping, Nonlinear Precoding, and Trellis Coding for Voiceband Telephone Channel Modems: With Emphasis on ITU-T Recommendation V.34 (The ... Series in Engineering and Computer Science). (Springer, 2002). Page 13.

Then

$$\mathbf{c}(\mathbf{a}) = [a_{L-1}, \ldots, a_0]\mathbf{G} = \mathbf{a}\mathbf{G} \tag{1.31}$$

The two-way partitioning process is illustrated in the Figure 1.6 for the case $L = 2$ by a partition tower and a partition tree.



(a) Partition Tower (b) Partition Tree

Figure 1.6. Illustration of an Ungerboeck Labeling by a Partition Tower and by a Partition Tree

Partitions of a two-dimensional Mod-2 lattice Λ with respect to $\Lambda' = 2\mathbb{Z}^2$ have the special property that the labeling can be made linear. This is important for one implementation of *trellis shaping*, a technique that will be presented in Chapter 4. It causes the trellis code to be linear when the convolutional code selecting the labels is linear. Suppose that λ_1 and λ_2 are lattice points in Λ. Then there are two vectors λ' and λ'' in $\Lambda' = 2\mathbb{Z}^2$ and two binary label vectors $\mathbf{a}_1$ and $\mathbf{a}_2$ such that

$$\lambda_1 = \lambda' + \mathbf{c}(\mathbf{a}_1) = \lambda' + \mathbf{a}_1\mathbf{G} \tag{1.32}$$

and

$$\lambda_2 = \lambda'' + \mathbf{c}(\mathbf{a}_2) = \lambda'' + \mathbf{a}_2\mathbf{G} \tag{1.33}$$

Adding these two vectors gives

$$\lambda_1 + \lambda_2 = \lambda' + \lambda'' + (\mathbf{a}_1 + \mathbf{a}_2)\mathbf{G} \tag{1.34}$$

Notice that $\lambda' + \lambda''$ is another point in $2\mathbb{Z}^2$. The elements of $\mathbf{a}_1 + \mathbf{a}_2$ must be 0, 1, or 2 since $\mathbf{a}_1$ and $\mathbf{a}_2$ are both binary vectors. A vector $\mathbf{v}$ that has 0's in the positions where $\mathbf{a}_1 + \mathbf{a}_2$ has 0 or 1 and has 2's in the positions where $\mathbf{a}_1 + \mathbf{a}_2$ is 2 can be subtracted from $\mathbf{a}_1 + \mathbf{a}_2$ to reduce it to a binary vector. The components of the resulting binary vector are just the mod-2 sums of the components of the two a's. This can be summarized by the following equation:

$$\mathbf{a}_1 + \mathbf{a}_2 = (\mathbf{a}_1 \oplus \mathbf{a}_2) + \mathbf{v} \tag{1.35}$$

Since $\mathbf{v}$ contains only 0's and 2's, the point $\mathbf{vG}$ is an element of $2\mathbf{Z}^2$ and $\lambda''' = \lambda' + \lambda'' + \mathbf{vG}$ must also be a point in $2\mathbf{Z}^2$. Thus

$$\lambda_3 = \lambda_1 + \lambda_2 = \lambda' + \lambda'' + \mathbf{vG} + (\mathbf{a}_1 \oplus \mathbf{a}_2)\mathbf{G} = \lambda''' + (\mathbf{a}_1 \oplus \mathbf{a}_2)\mathbf{G} \quad (1.36)$$

So, the coset label for the sum of the two lattice points is the mod-2 sum of the coset labels, that is,

$$\mathbf{a}_3 = \mathbf{a}_1 \oplus \mathbf{a}_2 \quad (1.37)$$

EXAMPLE 1.14

Consider the partition chain $\mathbf{Z}^2/R\mathbf{Z}^2/2\mathbf{Z}^2$. From Figures 1.2 and 1.3, we can see that

$$R\mathbf{Z}^2 = \bigcup_{a_1 \in \{0,1\}} \{2\mathbf{Z}^2 + a_1[1,1]\} \quad (1.38)$$

Comparing Figures 1.1 and 1.2, we see that

$$\mathbf{Z}^2 = \bigcup_{a_0 \in \{0,1\}} \{R\mathbf{Z}^2 + a_0[1,0]\} \quad (1.39)$$

Combining the two equations above gives

$$\begin{aligned} \mathbf{Z}^2 &= \bigcup_{a_1, a_0 \in \{0,1\}} \{2\mathbf{Z}^2 + a_1[1,1] + a_0[1,0]\} \\ &= \bigcup_{a_1, a_0 \in \{0,1\}} \left\{2\mathbf{Z}^2 + [a_1, a_0]\begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}\right\} \end{aligned}$$

So,

$$\mathbf{g}_1 = [1,1], \quad \mathbf{g}_0 = [1,0], \quad \text{and} \quad \mathbf{G} = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}$$

■

1.5 Fundamental Regions and Volumes, and Voronoi Regions

The points in a lattice Λ occur with a certain density, that is, number of points per unit volume in the N-dimensional space. The reciprocal of the density is the volume that can be associated with each lattice point and is called the *fundamental volume* for the lattice. The fundamental volume is denoted by $V(\Lambda)$. A *fundamental region* $\mathbb{R}(\Lambda)$ of the lattice can be loosely defined as any region that contains exactly one lattice point and has a volume equal to the fundamental volume. Fundamental regions are not unique. Several

Steven A. Tretter. Constellation Shaping, Nonlinear Precoding, and Trellis Coding for Voiceband Telephone Channel Modems: With Emphasis on ITU-T Recommendation V.34 (The ... Series in Engineering and Computer Science). (Springer, 2002). Page 15.

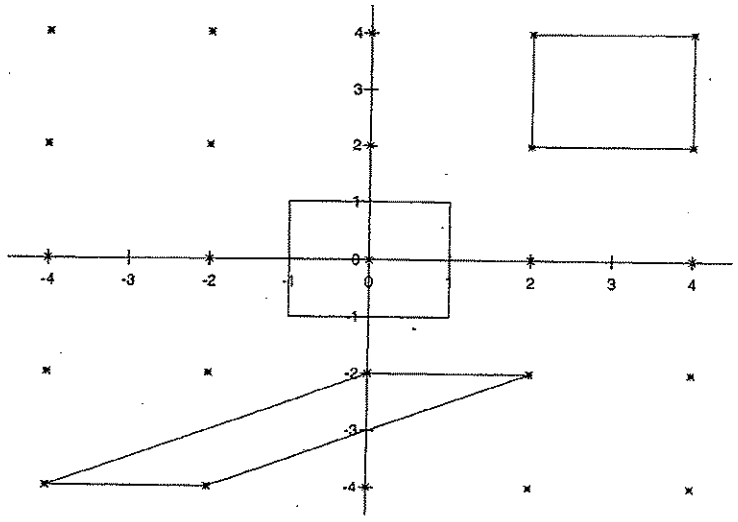


Figure 1.7. Some Fundamental Regions for the Lattice $2\mathbf{Z}^2$

fundamental regions for the lattice $2\mathbf{Z}^2$ are shown in Figure 1.7. Notice that each fundamental region has the same volume.

A more formal definition will now be given. Let $\mathbf{r}$ be a point in the N-dimensional space $\mathbf{R}^N$ of N-tuples with real coordinates and let $\boldsymbol{\lambda}$ be a point in the lattice $\boldsymbol{\Lambda}$. The set of points $\{\mathbf{r}+\boldsymbol{\lambda}|\boldsymbol{\lambda} \in \boldsymbol{\Lambda}\}$, which is the lattice $\boldsymbol{\Lambda}$ translated by the vector $\mathbf{r}$ and is sometimes denoted by $\mathbf{r} + \boldsymbol{\Lambda}$, is called an equivalence class of $\mathbf{r}$ in $\mathbf{R}^N$ modulo $\boldsymbol{\Lambda}$. Its points are said to be congruent modulo $\boldsymbol{\Lambda}$. A fundamental region of $\boldsymbol{\Lambda}$ is defined to be a region of points in $\mathbf{R}^N$ that contains one point from each equivalence class of $\mathbf{R}^N$ modulo $\boldsymbol{\Lambda}$.

A region which contains all the points of $\mathbf{R}^N$ closer to the origin (which is always a lattice point) than to any other lattice point is called a Voronoi region. Except for some boundary points, a Voronoi region is a fundamental region. The square enclosing the origin in Figure 1.7 is the Voronoi region for the lattice $2\mathbf{Z}^2$.

The regions that are formed when a fundamental region is translated by different lattice points do not overlap. The union of all such translations is $\mathbf{R}^N$. The collection of these non-overlapping regions is said to form a *tessellation* of $\mathbf{R}^N$.

1.5.1 Formula for the Fundamental Volume

The fundamental volume $V(\boldsymbol{\Lambda})$ for an N-dimensional lattice $\boldsymbol{\Lambda}$ in an N-dimensional space with a generator matrix $\mathbf{G}$ is

$$V(\boldsymbol{\Lambda}) = |\det \mathbf{G}| \tag{1.40}$$

First, consider the 2-dimensional case with basis vectors

$$\mathbf{x}_1 = [x_{1,1}, x_{1,2}] \tag{1.41}$$
$$\mathbf{x}_2 = [x_{2,1}, x_{2,2}] \tag{1.42}$$

and generator matrix

$$\mathbf{G} = \begin{bmatrix} x_{1,1} & x_{1,2} \\ x_{2,1} & x_{2,2} \end{bmatrix} \tag{1.43}$$

The basis vectors also can be represented in the complex plane by

$$x_1 = x_{1,1} + j\,x_{1,2} = |x_1|e^{j\theta_1} \tag{1.44}$$
$$x_2 = x_{2,1} + j\,x_{2,2} = |x_2|e^{j\theta_2} \tag{1.45}$$

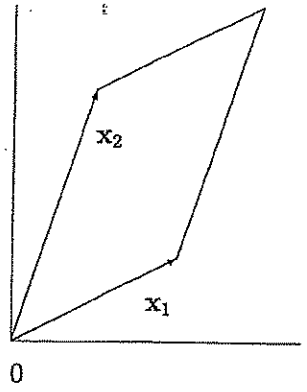


Figure 1.8. Parallelogram Formed by the Basis Vectors

These vectors form two adjacent edges of the parallelogram shown in Figure 1.8 which has area

$$V = |x_1||x_2| \sin|\theta_1 - \theta_2| \tag{1.46}$$

Notice that

$$x_2\bar{x}_1 = |x_1||x_2|e^{j(\theta_2-\theta_1)} \tag{1.47}$$

and

$$\Im m\{x_2\bar{x}_1\} = |x_1||x_2|\sin(\theta_2 - \theta_1) = x_{1,1}x_{2,2} - x_{1,2}x_{2,1} = \det \mathbf{G} \quad (1.48)$$

Thus, we see that $V(\Lambda) = |\det \mathbf{G}|$. The proof for arbitrary N can be done by induction.

If Λ is an N-dimensional lattice in an M-dimensional space where $M > N$, the square of the fundamental volume can be shown to be

$$V^2(\Lambda) = \det \mathbf{G}\mathbf{G}^t \quad (1.49)$$

1.5.2 Linear Transformations and the Fundamental Volume

Let Λ be an N-dimensional lattice with generator matrix $\mathbf{G}$ and $\mathbf{A}$ a non-singular $N \times N$ matrix. A new lattice $\tilde{\Lambda}$ can be formed with generator matrix $\tilde{\mathbf{G}} = \mathbf{GA}$. The new lattice consists of the set of points $\{\mathbf{xA}|\mathbf{x} \in \Lambda\} = \{\mathbf{zGA}|\mathbf{z} \in \mathbf{Z}\}$. The fundamental volume for the transformed lattice is

$$V(\tilde{\Lambda}) = |\det(\mathbf{GA})| = |\det(\mathbf{G})|\,|\det(\mathbf{A})| = V(\Lambda)\,|\det(\mathbf{A})| \quad (1.50)$$

In particular, suppose $\mathbf{A} = c\mathbf{I}$ so the new lattice is the original one scaled by the factor c. The new fundamental volume is

$$V(\bar{\Lambda}) = V(\Lambda)\,|\det(\mathbf{A})| = V(\Lambda)|c|^N \quad (1.51)$$

1.5.3 Fundamental Volume of a Sublattice

If Λ/Λ' is a lattice partition of order $L = |\Lambda/\Lambda'|$, then the fundamental volume of Λ' is L times that of Λ, that is

$$V(\Lambda') = |\Lambda/\Lambda'| \times V(\Lambda) \quad (1.52)$$

A heuristic proof of this fact is that Λ' is less dense than Λ by a factor of L, so its fundamental volume must be L times greater to fill N-space.

EXAMPLE 1.15 Fundamental Volumes for Some Lattices

The fundamental volume for $\mathbf{Z}^N$ is $V(\mathbf{Z}^N) = \det I_N = 1$. The partition $\mathbf{Z}^2/R\mathbf{Z}^2$ has order 2, so $V(R\mathbf{Z}^2) = 2$ according to (1.52). According to (1.40) and $\mathbf{G}$ in Example 1.3

$$V(R\mathbf{Z}^2) = \left|\det\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}\right| = 2 \quad (1.53)$$

as expected.

Since $|\mathbf{Z}^4/\mathbf{D}_4| = 2$, $V(\mathbf{D}_4) = 2$ according to (1.52). According to (1.40) and $\mathbf{G}$ in Example 1.6

$$V(\mathbf{D}_4) = \left|\det\begin{bmatrix} 2 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{bmatrix}\right| = 2 \quad (1.54)$$

as expected.

In Example 1.13 we saw that each partition in the chain $\mathbf{Z}^4/\mathbf{D}_4/R\mathbf{Z}^4$ has order two so that $\mathbf{Z}^4/R\mathbf{Z}^4$ has order 4 and $V(R\mathbf{Z}^4) = 4$. According to (1.40) and $\mathbf{R}_4$ in Example 1.7

$$V(R\mathbf{Z}^4) = \left|\det\begin{bmatrix} 1 & 1 & 0 & 0 \\ 1 & -1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & -1 \end{bmatrix}\right| = 4 \quad (1.55)$$

as expected. ∎

1.6 Point Spacing, Weight Distributions, and Theta Series

An important geometric property of a lattice is the minimum distance between lattice points. This parameter is a major factor at high signal-to-noise ratios in determining the theoretical error probability for a communication system using a signal constellation consisting of a finite set of points from the lattice. The squared Euclidean norm or energy of an n-tuple, $\mathbf{v}$, is defined as

$$\|\mathbf{v}\|^2 = \sum_{i=1}^{n} |v_i|^2 \quad (1.56)$$

The squared Euclidean distance, d^2, between any two points $\mathbf{x}$ and $\mathbf{y}$ is the squared norm of their difference, that is

$$d^2 = \|\mathbf{x} - \mathbf{y}\|^2 = \sum_{i=1}^{n} |x_i - y_i|^2 \quad (1.57)$$

When $\mathbf{x}$ and $\mathbf{y}$ are both points in a lattice, Λ, their difference is also a point in the lattice. Therefore, when searching for the minimum distance between lattice points, it suffices to just search for distances from the origin. This observation shows that the same set of distances is observed looking out from any point in the lattice. In other words, if a fixed lattice point is subtracted from all lattice points to translate the fixed point to the origin, the resulting set of points is

identical to the original untranslated lattice. The minimum squared distance, $d^2_{\min}(\Lambda)$, of a lattice Λ is defined to be

$$d^2_{\min}(\Lambda) = \min_{\mathbf{x},\mathbf{y}\in\Lambda,\mathbf{x}\neq\mathbf{y}} \|\mathbf{x}-\mathbf{y}\|^2 = \min_{\mathbf{x}\in\Lambda,\mathbf{x}\neq 0} \|\mathbf{x}\|^2 \tag{1.58}$$

Since a lattice contains a discrete set of points, the distances from the origin are a discrete set of numbers. Let the number of points at distance d from the origin in a lattice Λ be denoted by $N_d(\Lambda)$. The set of pairs of numbers $(d, N_d(\Lambda))$ is called the *weight distribution* of the lattice. The weight distribution is also the distribution of distances from any fixed point in the lattice. The number of points with norm $d_{\min}(\Lambda)$, which are the ones closest to the origin, is called the *kissing number*, *multiplicity*, or *error coefficient* for the lattice. We will denote the kissing number by $N_{\min}(\Lambda)$. Each lattice point has $N_{\min}(\Lambda)$ nearest neighbors.

Sometimes it is convenient to represent the weight distribution for a lattice by a series called the *theta series*. The theta series for a lattice Λ is defined to be

$$\Theta_\Lambda(q) = \sum_{\mathbf{x}\in\Lambda} q^{\|\mathbf{x}\|^2} \tag{1.59}$$

$$= \sum_{d^2=0}^{\infty} N_d(\Lambda) q^{d^2} \tag{1.60}$$

EXAMPLE 1.16 Theta Series for $\mathbf{Z}$

There are two integers at distance d from 0, d and $-d$. Therefore, $d_{\min}(\mathbf{Z}) = 1$ and $N_{\min}(\mathbf{Z}) = 2$. The theta series is

$$\Theta_{\mathbf{Z}}(q) = 1 + 2\sum_{d=1}^{\infty} q^{d^2} = 1 + 2q + 2q^4 + 2q^9 + \cdots \tag{1.61}$$

This function is known as the Jacobi theta function $\theta_3(q)$. ■

EXAMPLE 1.17 Theta Series for $\mathbf{Z}^N$

The N unit vectors $(0,\ldots,0,1,0,\ldots,0)$ and their negatives are the integer N-tuples closest to the origin. Therefore, $d_{\min}(\mathbf{Z}^N) = 1$ and $N_{\min}(\mathbf{Z}^N) = 2N$. The theta series is

$$\Theta_{\mathbf{Z}^N}(q) = \sum_{\mathbf{x}\in\mathbf{Z}^N} q^{\|\mathbf{x}\|^2} = \sum_{x_1=-\infty}^{\infty}\sum_{x_2=-\infty}^{\infty}\cdots\sum_{x_N=-\infty}^{\infty} q^{x_1^2+x_2^2+\cdots+x_N^2}$$

$$= \prod_{i=1}^{N}\sum_{x_i=-\infty}^{\infty} q^{x_i^2} = \Theta_{\mathbf{Z}}^N(q) \tag{1.62}$$

In particular, consider the case when $N = 2$. Then the first few terms of the theta series are

$$\Theta_{\mathbf{Z}^2}(q) = \Theta_{\mathbf{Z}}^2(q) = 1 + 4q + 4q^2 + 4q^4 + 8q^5 + 4q^9 + \cdots \tag{1.63}$$

The four points with $d^2 = 1$ are the $2N = 4$ smallest points $(\pm1, 0)$ and $(0, \pm1)$, the four points with $d^2 = 2$ are $(\pm1, \pm1)$, the four points with $d^2 = 4$ are $(\pm2, 0)$ and $(0, \pm2)$, the eight points with $d^2 = 5$ are $(\pm2, \pm1)$ and $(\pm1, \pm2)$, and so on. ■

EXAMPLE 1.18 Theta Series for $\mathbf{D}_N$

Remember that $\mathbf{D}_N$ is the set of integer N-tuples whose sum of components is even or, equivalently, contain an even number of odd components. The points closest to the origin have the form

$$\mathbf{x} = (0,\ldots,0,\pm1,0,\ldots,0,\pm1,0,\ldots,0)$$

The positions with ±1 can be selected $\binom{N}{2}$ ways and each of these pairs can be one of the four pairs, $(1,1)$, $(1,-1)$, $(-1,1)$, or $(-1,-1)$. Therefore, $d^2_{\min}(\mathbf{D}_N) = 2$ and $N_{\min}(\mathbf{D}_N) = 4\binom{N}{2}$.

The theta series for $\mathbf{D}_N$ can be expressed in terms of the theta series for $\mathbf{Z}^N$. The derivation depends on the following facts:

1 The square of an even number is even while the square of an odd number is odd.

2 The sum of any number of even integers is even and the sum of an even number of odd integers is even.

3 The sum of an odd number of odd integers is odd.

4 From the previous items it follows that the sum of an even number of squares of odd integers is even while the sum of an odd number of squares of odd integers is odd.

Based on these facts and the definition of $\mathbf{D}_N$, it follows that the squared norm of any element of $\mathbf{D}_N$ is an even integer. Also, $\mathbf{Z}^N/\mathbf{D}_N$ is a two way partition of $\mathbf{Z}^N$ so $\mathbf{Z}^N$ is the union of $\mathbf{D}_N$ and its coset consisting of points whose squared norm is odd. The theta series is the series for $\mathbf{Z}^N$ with the odd powers of q deleted. Therefore

$$\Theta_{\mathbf{D}_N}(q) = \frac{1}{2}[\Theta_{\mathbf{Z}^N}(q) + \Theta_{\mathbf{Z}^N}(-q)]$$

$$= \frac{1}{2}\left[\sum_{d=0}^{\infty} N_d(\mathbf{Z}^N) q^{d^2} + \sum_{d=0}^{\infty} N_d(\mathbf{Z}^N)(-q)^{d^2}\right]$$

$$= \frac{1}{2}[\Theta_{\mathbf{Z}}^N(q) + \Theta_{\mathbf{Z}}^N(-q)] \quad (1.64)$$

As a particular example, the theta series for $\mathbf{D}_4$ is

$$\begin{aligned}\Theta_{\mathbf{D}_4}(q) &= \frac{1}{2}[\Theta_{\mathbf{Z}}^4(q) + \Theta_{\mathbf{Z}}^4(-q)] \\ &= 1 + 24q^2 + 24q^4 + 96q^6 + 24q^8 + 144q^{10} + \cdots \quad (1.65)\end{aligned}$$

See page 119 of Conway and Sloane [12] for a weight distribution table for $\mathbf{D}_4$ up to squared norm 118. ■

Once again consider the partition chain $\mathbf{Z}^4/\mathbf{D}_4/R\mathbf{Z}^4/R\mathbf{D}_4/2\mathbf{Z}^4/2\mathbf{D}_4$ presented in Example 1.13. Each partition in the chain is two-way. We have seen that $d_{\min}^2(\mathbf{Z}^4) = 1$ and $d_{\min}^2(\mathbf{D}_4) = 2$. The rotation operator R increases squared distances by a factor of 2 so $d_{\min}^2(R\mathbf{Z}^4) = 2$ and $d_{\min}^2(R\mathbf{D}_4) = 4$. Continuing in this manner, we see that this partition chain has the minimum squared distances 1/2/2/4/4/8.

1.7 Fundamental Coding Gain

The *fundamental coding gain* of an N-dimensional lattice Λ is defined as

$$\gamma_c(\Lambda) = \frac{d_{\min}^2(\Lambda)}{V^{2/N}(\Lambda)} \quad (1.66)$$

This parameter is a measure of the density of the lattice Λ when its minimum distance is normalized to 1 and the volume is referred to two dimensions. Notice that the numerator has units of length^2. The volume $V(\Lambda)$ has units of length^N so the denominator also has units of length^2. Therefore, $\gamma_c(\Lambda)$ is a dimensionless quantity.

The scaled lattice $d_{\min}^{-1}(\Lambda)\,\Lambda$ has minimum squared distance 1 and, according to (1.51), fundamental volume $d_{\min}^{-N}(\Lambda)\,V(\Lambda)$. Also, $V(\mathbf{Z}^N) = 1$. Therefore, the fundamental coding gain can be written as

$$\gamma_c(\Lambda) = \left[\frac{V(\mathbf{Z}^N)}{V\left(d_{\min}^{-1}(\Lambda)\,\Lambda\right)}\right]^{2/N} \quad (1.67)$$

Thus the fundamental coding gain gives a measure of the density of a lattice Λ scaled to have unity minimum distance relative to the density of $\mathbf{Z}^N$.

The reason why $\gamma_c(\Lambda)$ is called a coding gain is explained in detail in Section 2.5.1. Briefly, if a signal constellation with M points is chosen by selecting the M points closest to the origin from a lattice Λ, the coding gain is the ratio of the average power of an M-point baseline constellation selected from $\mathbf{Z}^N$ to the average power of the constellation selected from Λ when it is scaled to have

minimum distance 1 which is the same as for $\mathbf{Z}^N$. At high signal-to-noise ratios, the theoretical symbol error probability for a constellation depends primarily on the minimum distance. Therefore, the fundamental coding gain is a measure of the reduction of signal power achieved by using the lattice Λ instead of $\mathbf{Z}^N$ for the same symbol error probability.

Two additional properties of the fundamental coding gain are:

(a) The fundamental coding gain is invariant to scaling of a lattice. Let $\Lambda_1 = c\Lambda$. Then $d_{\min}^2(\Lambda_1) = c^2 d_{\min}^2(\Lambda)$ and $V(\Lambda_1) = |c|^N V(\Lambda)$. Therefore

$$\gamma_c(\Lambda_1) = \frac{c^2 d_{\min}^2(\Lambda)}{[|c|^N V(\Lambda)]^{2/N}} = \gamma_c(\Lambda)$$

(b) The fundamental coding gain is invariant to any scaled orthogonal transformation of a lattice. Let $\mathbf{A}$ be and $N \times N$ nonsingular matrix with the property $\mathbf{A}\mathbf{A}^t = c^2\mathbf{I}$ so that $\det(\mathbf{A}\mathbf{A}^t) = \det^2(\mathbf{A}) = c^{2N}$ or $|\det(\mathbf{A})| = |c|^N$. Let the transformed lattice Λ_1 consist of the points $\{\mathbf{x}\mathbf{A} | \mathbf{x} \in \Lambda\}$. According to (1.50), $V(\Lambda_1) = V(\Lambda)|\det\mathbf{A}| = V(\Lambda)|c|^N$. Let $\mathbf{x}_1 = \mathbf{x}\mathbf{A}$ where $\mathbf{x} \in \Lambda$. Then $\|\mathbf{x}_1\|^2 = \mathbf{x}\mathbf{A}\mathbf{A}^t\mathbf{x}^t = c^2\|\mathbf{x}\|^2$ so $d_{\min}^2(\Lambda_1) = c^2 d_{\min}^2(\Lambda)$. Thus the coding gain for Λ_1 is

$$\gamma_c(\Lambda_1) = \frac{|c|^2 d_{\min}^2(\Lambda)}{[|c|^N V(\Lambda)]^{2/N}} = \gamma_c(\Lambda)$$

These properties show that lattices that are the same except for scaling and rotation have the same coding gains.

EXAMPLE 1.19 Fundamental Coding Gains for $\mathbf{Z}^N$ and $R\mathbf{Z}^N$

The lattice $\mathbf{Z}^N$ has $d_{\min}(\mathbf{Z}^N) = 1$ and $V(\mathbf{Z}^N) = 1$. Therefore, $\gamma_c(\mathbf{Z}^N) = 1$ or 0 dB.

Now assume N is even. The lattice $R\mathbf{Z}^N$ is the lattice Z^N transformed by a matrix of the form

$$\mathbf{A} = \begin{bmatrix} \mathbf{R} & 0 & 0 & 0 \\ 0 & \mathbf{R} & 0 & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & \mathbf{R} \end{bmatrix} \quad (1.68)$$

where

$$\mathbf{R} = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

It is easy to see that $\mathbf{A}\mathbf{A}^t = 2\mathbf{I}$ so $\mathbf{A}$ is a scaled orthogonal matrix with $c^2 = 2$. As shown in Property (b) above, $V(R\mathbf{Z}^N) = |c|^N V(\mathbf{Z}^N) = 2^{N/2}$. Also,

$d^2_{\min}(R\mathbf{Z}^N) = 2$. Therefore, $\gamma_c(R\mathbf{Z}^N) = 2/(2^{N/2})^{2/N} = 1$ which is exactly what Property (b) requires. ∎

EXAMPLE 1.20 Fundamental Coding Gain for $\mathbf{D}_4$

We have seen that $d^2_{\min}(\mathbf{D}_4) = 2$ and $V(\mathbf{D}_4) = 2$. Therefore, $\gamma_c(\mathbf{D}_4) = 2/2^{2/4} = \sqrt{2}$ or $10\log_{10}\sqrt{2} = 1.505$ dB. Therefore, $\mathbf{D}_4$ is denser than $\mathbf{Z}^4$ by the factor $\sqrt{2}$. ∎

Chapter 2

PERFORMANCE MEASURES FOR MULTIDIMENSIONAL CONSTELLATIONS

2.1 Introduction

In general, an N-dimensional signal constellation $\mathbf{C}$ is a finite set of points selected from an N-dimensional space. The number of points in the constellation is denoted by $|\mathbf{C}|$ and is called the *size* of $\mathbf{C}$. For ease of implementation and compatibility with trellis coded modulation (TCM), the constellation points are usually selected from an N-dimensional lattice $\mathbf{\Lambda}$ (or a translation of $\mathbf{\Lambda}$) and lie in a finite region $\mathbb{R}$ of N-space. This type of constellation is sometimes called a *lattice code* and will be denoted by $\mathbf{C}(\mathbf{\Lambda}, \mathbb{R})$.

Quadrature amplitude modulation (QAM) is used by many modems operating over linear time-invariant channels with high signal-to-noise ratios (SNR) like twisted-pair local loop telephone lines. They use signal constellations with multiple points selected from two or higher dimensional lattices. QAM is a two-dimensional (2D) transmission scheme but points from an $N = 2L$ dimensional lattice can be sent by transmitting a sequence of $L = N/2$ two-dimensional symbols. Prior to the invention of trellis coding, the V series QAM modems used 2D constellations with 2^β points to transmit an integer number of bits β per symbol (or per two dimensions). Values of β from 2 to 4 corresponding to constellations of size 4 to 16 were used depending on the desired data rate. Trellis coding was introduced in the V.32 [33] recommendation where the constellation size was doubled to $2^{\beta+1}$ to transmit an integer number of bits β per 2D symbol but the signal constellation was still two-dimensional. V.32 uses a constellation with 32 points to transmit at 9600 bits/sec using 4 data bits and one redundant code bit per symbol with a symbol rate of 2400 symbols/sec. Not long afterwards, V.32 was extended to V.32bis to allow TCM transmission at 14,400 bits/sec with a 128-point 2D constellation using 6 data bits and one redundant code bit per symbol with a symbol rate of 2400 symbols/sec.

The V.34 recommendation [34] standardized a number of modem innovations. One was trellis codes with 16, 32, or 64 states using four-dimensional symbols. Some previous proprietary modems had used multidimensional lattice block codes and four or eight-dimensional trellis codes. The V.34 recommendation also specified a technique called shell mapping for constellation shaping that operates on blocks of four 4D symbols (or 16 dimensions). Shell mapping is discussed in Chapter 8. The combination of 4D TCM and shell mapping allows the transmission of a fractional number of bits β per 2D symbol with data rates ranging from 2400 bits/sec to 33,600 bits/sec in steps of 2400 bits/sec. At the highest data rates, the V.34 2D symbols can take on over 1000 values with the point probabilities approximating a two-dimensional circular Gaussian probability density function with its peak at the origin. Therefore, the points near the origin which have lower energy are more likely to occur. All ITU-T modems previous to V.34 use points with equal likelihood in two dimensions. A result in information theory proves that the input to an additive Gaussian noise channel must have a Gaussian distribution to achieve channel capacity [27]. The V.34 modem takes a large step towards this goal with the shell mapping.

In this chapter, some quantitative measures of the properties of multidimensional constellations that can be used to compare the merits of different constellation designs are presented. A constellation called the N-cube grid consisting of the points of the N-dimensional integer lattice $\mathbf{Z}^N$ contained in an N-cube is used as a baseline for comparison throughout the chapter. First, the constellation figure of merit, CFM(C), is defined in Section 2.2. The CFM allows constellations with the same minimum distance between points to be compared in terms of power efficiency in two dimensions. The continuous approximation for constellations of large size is introduced. This technique approximates the discrete constellation point distribution by a continuous probability density function over a region just containing the constellation and allows simplified approximate computations with constellations of large size.

The concept of a constituent 2D constellation is introduced in Section 2.3 and the constellation expansion ratio, CER(C), is defined. Basically, the constituent 2D constellation is obtained by projecting the multidimensional constellation onto two dimensions and the CER is the ratio of the size of the constituent 2D constellation to the size of the constituent 2D constellation induced by the baseline unshaped N-cube grid. In Section 2.6, it is shown how the CER can be decomposed into the product of a coding factor and a shaping factor.

The peak-to-average power ratio, PAR, is discussed in Section 2.4 and several examples are presented. In particular, it is shown that the PAR for an N-sphere becomes infinite with N. It is shown in Section 2.7 how the PAR can be factored into the product of the PAR for the constituent 2D region with a uniform point distribution, a biasing gain, and the shaping constellation expansion ratio. This formula is useful in designing constellations with a desired shaping gain while maintaining an acceptable PAR.

In Section 2.5 it is shown how the ratio of the CFM for constellation to the baseline CFM can be factored into the product of a term, $\gamma_c(\mathbf{\Lambda})$, called the coding gain and a term, $\gamma_s(\mathbb{R})$, called the shaping gain. The coding gain has already been defined in Section 1.7. It is a measure of the improvement achieved by using a lattice denser than the N-cube grid. The shaping gain is a measure of the improvement achieved by using a constellation more spherically shaped than an N-cube. It is shown that the maximum achievable shaping gain is $\pi e/6 = 1.53$ dB and that an N-sphere reaches this limit as N becomes large. In addition, it is demonstrated that the constituent 2D probability density function must be a circular Gaussian density to reach this shaping limit.

The peak-to-average ratio and constellation expansion ratio become infinite as the ultimate achievable shaping gain of 1.53 dB is approached. However, points with large power occur with low probability so we would expect intuitively that eliminating some large points from the constellation should have little effect on the shaping gain. In Section 2.8 the effects of confining the constituent 2D constellation to the interior of a circle to limit the peak power is investigated. Some formulas for optimum tradeoffs of shaping gain with peak-to-average ratio and shaping constellation expansion ratio are presented. It is observed that over 1 dB of shaping gain can be achieved with reasonable PAR and CER.

Finally, well deserved credit must be given to Forney and Wei for their highly original and now classic paper "Multidimensional Constellations—Part I: Introduction, Figures of Merit, and Generalized Cross Constellations" [25]. This chapter is heavily based on their work.

2.2 Constellation Figure of Merit and Symbol Error Probabilities

Bit error probability as a function of signal-to-noise ratio (SNR) is a common measure of the performance of a digital communication system. The bit error probability is typically related to the symbol error probability by a scale factor in the order of unity. Systems using the same bandwidth and data rate can be ranked in terms of their error probabilities at identical SNR's. The one with the lower error probability is considered to be better. At high SNR the symbol error probability is determined primarily by the probability of making an error to one of the closest neighbors of the transmitted constellation point. This probability is a function of the minimum distance between points, $d_{\min}$, and the noise variance. In this section, a parameter called the constellation figure of merit, CFM(C), is defined which allows different constellations with the same minimum distance between points to be compared in terms of signal power. The constellation figure of merit and symbol error probability in terms of SNR

and CFM for two examples, the one-dimensional PAM constellation and the two-dimensional $M \times M$ square grid, are derived. Finally, it is shown how the discrete constellation point probability distribution can be approximated by a continuous distribution over a region closely bounding the constellation to obtain close approximate results for constellations with a large number of points. This continuous approximation is used throughout the rest of the chapter.

2.2.1 Normalized Bit Rate and Average Power

Suppose a QAM modem uses an N-dimensional constellation $\mathbf{C}$ that is a subset of an N-dimensional lattice $\mathbf{\Lambda}$ to transmit b bits per N-dimensional symbol where each N-dimensional symbol is transmitted as a sequence of $N/2$ 2D symbols.. Each 2D symbol period is sometimes called a *baud*. In this book only signal constellations selected from an N-dimensional lattice or its translation will be considered so the points are regularly spaced and have the same number of nearest neighbors except at the constellation boundaries. The size of $\mathbf{C}$ must be at least 2^b. Since QAM symbols are two dimensional, we will normalize various constellation parameters to two dimensions. For example, the *normalized bit rate* is defined to be

$$\beta = \frac{b}{N/2} = \frac{2b}{N} \text{ bits per baud} \tag{2.1}$$

The average power of an N-dimensional constellation $\mathbf{C}$ assuming that points are used with equal probability is

$$P(\mathbf{C}) = \frac{1}{|\mathbf{C}|} \sum_{\mathbf{x} \in \mathbf{C}} \|\mathbf{x}\|^2 \tag{2.2}$$

and the power normalized to two dimensions is

$$\mathbb{P}(\mathbf{C}) = \frac{P(\mathbf{C})}{N/2} = \frac{2P(\mathbf{C})}{N} \tag{2.3}$$

The minimum squared distance $d^2_{\min}(\mathbf{C})$ for the constellation is

$$d^2_{\min}(\mathbf{C}) = \min_{\mathbf{x},\mathbf{y} \in \mathbf{C}; \mathbf{x} \neq \mathbf{y}} \|\mathbf{x} - \mathbf{y}\|^2 \tag{2.4}$$

This is usually the same as $d^2_{\min}(\mathbf{\Lambda})$. The average power $P(\mathbf{C})$ is proportional to $d^2_{\min}(\mathbf{C})$ since the constellation points are regularly spaced.

2.2.2 Definition of CFM and Examples

The symbol error probability for a constellation used over an additive white Gaussian noise channel is a function of the ratio of the minimum squared distance to the noise variance. Two constellations with the same $d_{\min}$ will have

nearly the same error probability but the one with the smaller $\mathbb{P}(\mathbf{C})$ will be more efficient in terms of power use. The constellation with the smaller $\mathbb{P}(\mathbf{C})$ can be scaled up to increase $d_{\min}$ resulting in a smaller error probability for the same average transmitted signal power. N-dimensional constellations with the same normalized bit rate β can be ranked by using a parameter called the *constellation figure of merit* which is defined as

$$\text{CFM}(\mathbf{C}) = \frac{d^2_{\min}(\mathbf{C})}{\mathbb{P}(\mathbf{C})} \tag{2.5}$$

If $\mathbf{C}$ is scaled by the factor μ resulting in the constellation $\mu\mathbf{C}$, then $d^2_{\min}(\mu\mathbf{C}) = \mu^2 d_{\min}(\mathbf{C})$ and $\mathbb{P}(\mu\mathbf{C}) = \mu^2\mathbb{P}(\mathbf{C})$ so $\text{CFM}(\mu\mathbf{C}) = \text{CFM}(\mathbf{C})$. In words, the constellation figure of merit is invariant to constellation scaling. The CFM can also be expressed as

$$\text{CFM}(\mathbf{C}) = \frac{1}{\mathbb{P}(\mathbf{C}/d_{\min}(\mathbf{C}))} \tag{2.6}$$

The scaled constellation $\mathbf{C}/d_{\min}(\mathbf{C})$ has minimum squared distance equal to 1.

EXAMPLE 2.1 One-Dimensional PAM Constellation

A one-dimensional pulse amplitude modulation (PAM) constellation $\mathbf{C}_a$ for a data rate of b bits per symbol is the set of $M = 2^b$ points

$$c_i = d\left(\frac{1}{2} + i\right) \text{ for } i = -\frac{M}{2}, \ldots, -1, 0, 1, \ldots, \frac{M}{2} - 1 \tag{2.7}$$

This constellation is formed by taking the M points closest to the origin from the translated lattice $\mathbf{Z}+0.5$ and scaling them by d. The points in $\mathbf{Z}+0.5$ are called *half integers*. Observe that the points in $\mathbf{C}_a$ are $\pm d/2, \pm 3d/2, \ldots \pm (M-1)d/2$ which are symmetric about the origin. The normalized data rate is $\beta = 2b$ bits per 2D and $d_{\min}(\mathbf{C}_a) = d$. The average power is

$$P(\mathbf{C}_a) = d^2 \frac{2}{M} \sum_{i=0}^{\frac{M}{2}-1} \left(\frac{1}{2} + i\right)^2 = (M^2 - 1)\frac{d^2}{12} = (2^\beta - 1)\frac{d^2}{12} \tag{2.8}$$

and the normalized average power per two dimensions is

$$\mathbb{P}(\mathbf{C}_a) = 2P(\mathbf{C}_a) = (2^\beta - 1)\frac{d^2}{6} \tag{2.9}$$

The constellation figure of merit is

$$\text{CFM}(\mathbf{C}_a) = \frac{d^2_{\min}(\mathbf{C}_a)}{\mathbb{P}(\mathbf{C}_a)} = \frac{6}{M^2 - 1} = \frac{6}{2^\beta - 1} \tag{2.10}$$

Suppose the symbol c_i is transmitted and $r = c_i + v$ is received where v is a Gaussian random variable with zero mean and variance σ^2. The optimum decision boundaries are the points half way between the constellation points. In computing the symbol error probability, there are two situations to consider. One is for the two symbols at the outer edges of the constellation, $\pm d(M-1)/2$. For the positive edge symbol $d(M-1)/2$ a correct decision is made if $v > -d/2$. The probability of this event is

$$\begin{aligned} Pr\left(\text{correct} \,|\, i = (M-2)/2\right) &= Pr\left(v > -d/2\right) \\ &= \int_{-d/2}^{\infty} \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{t^2}{2\sigma^2}}\, dt = 1 - Q\left(\frac{d}{2\sigma}\right) \end{aligned} \tag{2.11}$$

where

$$Q(x) = \int_{x}^{\infty} \frac{1}{\sqrt{2\pi}} e^{-\frac{t^2}{2}}\, dt \tag{2.12}$$

is the Gaussian tail probability. The probability of error for this symbol is

$$\begin{aligned} Pr\left(\text{error} \,|\, i = (M-2)/2\right) &= 1 - Pr\left(\text{correct} \,|\, i = (M-2)/2\right) \\ &= Q\left(\frac{d}{2\sigma}\right) \end{aligned} \tag{2.13}$$

As a result of the symmetry of the Gaussian probability density function, the error probability for the negative edge symbol is the same. There are $M-2$ inner symbols. For each of them, a correct decision is made if $|v| < d/2$ and the probability of this event is

$$Pr\left(\text{correct} \,|\, i = 0\right) = \int_{-d/2}^{d/2} \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{t^2}{2\sigma^2}} = 1 - 2Q\left(\frac{d}{2\sigma}\right) \tag{2.14}$$

The probability of an error for the inner symbols is

$$Pr\left(\text{error} \,|\, i = 0\right) = 1 - Pr\left(\text{error} \,|\, i = 0\right) = 2Q\left(\frac{d}{2\sigma}\right) \tag{2.15}$$

Therefore, the average probability of error is

$$\begin{aligned} P_e &= \frac{2}{M} Pr\left(\text{error} \,|\, i = (M-2)/2\right) + \frac{M-2}{M} Pr\left(\text{error} \,|\, i = 0\right) \\ &= 2\frac{M-1}{M} Q\left(\frac{d}{2\sigma}\right) \end{aligned} \tag{2.16}$$

Solving (2.8) for d^2 in terms of $P(\mathbf{C}_a)$ and substituting into (2.15) gives the following formula for the error probability in terms of $\text{SNR} = P(\mathbf{C}_a)/\sigma^2$:

$$P_e = 2\frac{M-1}{M} Q\left[\left(\frac{3}{M^2-1}\frac{P(\mathbf{C}_a)}{\sigma^2}\right)^{1/2}\right]$$

$$= 2\frac{M-1}{M} Q\left[\left(\frac{\text{CFM}(\mathbf{C}_a)}{2}\frac{P(\mathbf{C}_a)}{\sigma^2}\right)^{1/2}\right] \tag{2.17}$$

EXAMPLE 2.2 $M \times M$ Square Grid

An $M^2 = 2^\beta$ point two-dimensional constellation $\mathbf{C}_b$ can be formed by taking the Cartesian product of an M-point PAM constellation with itself. That is

$$\mathbf{C}_b = \mathbf{C}_a \times \mathbf{C}_a = \{\mathbf{z} = (x, y) \,|\, x \in \mathbf{C}_a,\, y \in \mathbf{C}_a\} \tag{2.18}$$

The normalized bit rate is $\beta = 2\log_2 M$ bits per symbol and $d^2_{\min}(\mathbf{C}_b) = d^2$. The average and normalized average powers are

$$\begin{aligned} P(\mathbf{C}_b) = \mathbb{P}(\mathbf{C}_b) &= \mathrm{E}\{|\mathbf{z}|^2\} = \mathrm{E}\{x^2 + y^2\} = 2P(\mathbf{C}_a) \\ &= \frac{M^2-1}{6} d^2 = \frac{2^\beta - 1}{6} d^2 \end{aligned} \tag{2.19}$$

Therefore, the constellation figure of merit is

$$\text{CFM}(\mathbf{C}_b) = \frac{d^2_{\min}}{\mathbb{P}(\mathbf{C}_b)} = \frac{6}{M^2-1} = \frac{6}{2^\beta - 1} \tag{2.20}$$

which is the same as for PAM.

Three types of points must be considered to find the symbol error probability. The upper right-hand corner of $\mathbf{C}_b$ is shown in Figure 2.1 to illustrate these types. Suppose the point $\mathbf{z} = (x, y)$ is transmitted and $\mathbf{r} = \mathbf{z} + \mathbf{v} = (x, y) + (v_x, v_y)$ is received where v_x and v_y are independent, zero mean, Gaussian random variables each with variance σ^2. There are four corner points and the probability of a correct decision for each is

$$\begin{aligned} &Pr(\text{correct} \,|\, \text{corner}) = Pr(v_x > -d/2 \cap v_y > -d/2) \\ &= Pr(v_x > -d/2) Pr(v_y > -d/2) = \left[1 - Q\left(\frac{d}{2\sigma}\right)\right]^2 \end{aligned} \tag{2.21}$$

and the probability of an error is

$$\begin{aligned} Pr(\text{error} \,|\, \text{corner}) &= 1 - Pr(\text{correct} \,|\, \text{corner}) \\ &= 2Q\left(\frac{d}{2\sigma}\right) - Q^2\left(\frac{d}{2\sigma}\right) \end{aligned} \tag{2.22}$$

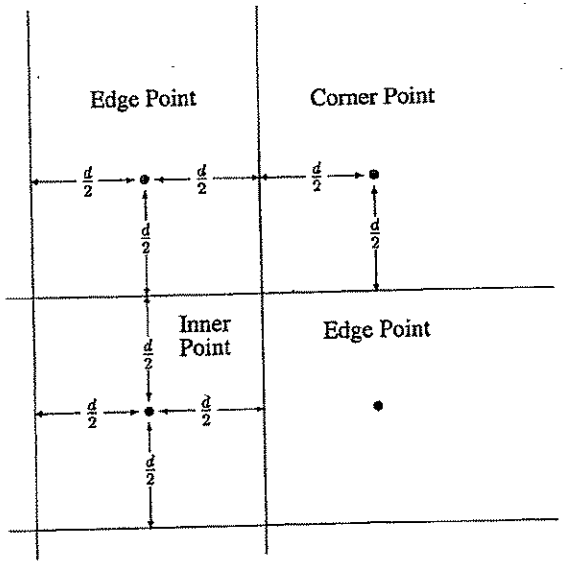


Figure 2.1. Three Types of Points in $\mathbf{C}_b$

There are $4(M-2)$ edge points. The probability of a correct decision for the upper edge point in Figure 2.1 is

$$\begin{aligned} Pr(\text{correct}\,|\,\text{edge}) &= Pr(|v_x| < d/2 \cap v_y > -d/2) \\ &= Pr(|v_x| < d/2)Pr(v_y > -d/2) \\ &= \left[1-2Q\left(\frac{d}{2\sigma}\right)\right]\left[1-Q\left(\frac{d}{2\sigma}\right)\right] \end{aligned} \tag{2.23}$$

and the error probability is

$$Pr(\text{error}\,|\,\text{edge}) = 1-Pr(\text{correct}\,|\,\text{edge}) = 3Q\left(\frac{d}{2\sigma}\right)-2Q^2\left(\frac{d}{2\sigma}\right) \tag{2.24}$$

There are $(M-2)^2$ inner points each with the probability of correct decision

$$\begin{aligned} Pr(\text{correct}\,|\,\text{inner}) &= Pr(|v_x| < d/2 \cap |v_y| < d/2) \\ &= Pr(|v_x| < d/2)Pr(|v_y| < d/2) = \left[1-2Q\left(\frac{d}{2\sigma}\right)\right]^2 \end{aligned} \tag{2.25}$$

and

$$Pr(\text{error}\,|\,\text{inner}) = 1-Pr(\text{correct}\,|\,\text{inner}) = 4Q\left(\frac{d}{2\sigma}\right)-4Q^2\left(\frac{d}{2\sigma}\right) \tag{2.26}$$

Putting these results together, the average symbol error probability is

$$\begin{aligned} P_e &= \frac{4}{M^2}Pr(\text{error}\,|\,\text{corner}) + \frac{4(M-2)}{M^2}Pr(\text{error}\,|\,\text{edge}) \\ &\quad + \frac{(M-2)^2}{M^2}Pr(\text{error}\,|\,\text{inner}) \\ &= 4\frac{M-1}{M}Q\left(\frac{d}{2\sigma}\right) - 4\left(\frac{M-1}{M}\right)^2 Q^2\left(\frac{d}{2\sigma}\right) \end{aligned} \tag{2.27}$$

The total noise power is $\mathrm{E}\left\{|\mathbf{v}|^2\right\} = 2\sigma^2$, so the SNR for this constellation is

$$\mathrm{SNR}_{M\times M} = \frac{\mathbb{P}(\mathbf{C}_b)}{2\sigma^2} \tag{2.28}$$

Solving (2.18) for d^2 in terms of $\mathbb{P}(\mathbf{C}_b)$, the error probability can be expressed as

$$\begin{aligned} P_e &= 4\frac{M-1}{M}Q\left[\left(\frac{3}{M^2-1}\frac{\mathbb{P}(\mathbf{C}_b)}{2\sigma^2}\right)^{1/2}\right] \\ &\quad - 4\left(\frac{M-1}{M}\right)^2 Q^2\left[\left(\frac{3}{M^2-1}\frac{\mathbb{P}(\mathbf{C}_b)}{2\sigma^2}\right)^{1/2}\right] \\ &= 4\frac{M-1}{M}Q\left[\left(\frac{\mathrm{CFM}(\mathbf{C}_b)}{2}\mathrm{SNR}_{M\times M}\right)^{1/2}\right] \\ &\quad - 4\left(\frac{M-1}{M}\right)^2 Q^2\left[\left(\frac{\mathrm{CFM}(\mathbf{C}_b)}{2}\mathrm{SNR}_{M\times M}\right)^{1/2}\right] \end{aligned} \tag{2.29}$$

EXAMPLE 2.3 N-Cube Grid

The generalization of the previous two examples is the N-dimensional N-cube grid $\mathbf{C}_c = \mathbf{C}_a^N$, the N-fold Cartesian product of the PAM constellation with itself. There are M^N points in this constellation so the bit rate is $b = N\log_2 M$ bits per symbol and the normalized bit rate is $\beta = 2\log_2 M$ bits per baud. The minimum distance is again $d_{\min}(\mathbf{C}_c) = d$ and the average power is

$$P(\mathbf{C}_c) = NP(\mathbf{C}_a) = N\frac{M^2-1}{12}d^2 \tag{2.30}$$

Therefore, the power normalized to two dimensions is

$$\mathbb{P}(\mathbf{C}_c) = 2P(\mathbf{C}_c)/N = \frac{M^2-1}{6}d^2 = \frac{2^\beta-1}{6}d^2 \tag{2.31}$$

and the constellation figure of merit is

$$\mathrm{CFM}(\mathbf{C}_c) = \frac{6}{M^2-1} = \frac{6}{2^\beta - 1} \tag{2.32}$$

This figure of merit is the same as for PAM and the $M \times M$ square grid. ■

PAM and the $M \times M$ square grid are both members of the N-cube grid family. Every member of this family with the same normalized bit rate β has the constellation figure of merit $6/(2^\beta - 1)$ which asymptotically approaches $6/2^\beta$ for large β. The N-cube grid family will be used as the baseline for comparing constellations. We will define the *baseline constellation figure of merit* for normalized bit rate β as

$$\mathrm{CFM}_\oplus(\beta) = \frac{6}{2^\beta} \tag{2.33}$$

and the *baseline normalized average power* as

$$\mathbb{P}_\oplus(\beta) = \frac{1}{\mathrm{CFM}_\oplus(\beta)} = \frac{2^\beta}{6} \tag{2.34}$$

The gain achieved by using a constellation $\mathbf{C}$ with normalized bit rate β instead of the baseline constellation is

$$\frac{\mathrm{CFM}(\mathbf{C})}{\mathrm{CFM}_\oplus(\beta)} = \frac{2^\beta d^2_{\min}(\mathbf{C})}{6\mathbb{P}(\mathbf{C})} = \frac{\mathbb{P}_\oplus(\beta)}{\mathbb{P}(\mathbf{C}/d_{\min}(\mathbf{C}))} \tag{2.35}$$

2.2.3 An Approximation to the Symbol Error Probability for Large Square QAM Constellations at High SNR

A voice-band telephone channel normally has a high SNR, typically 30 dB or more, and high-speed modems use constellations with many points. A simple formula for the symbol error probability of an $M \times M$ square grid constellation with a large number of points used over an additive Gaussian noise channel with high SNR will now be derived. Let the number of constellation points, $M^2 = 2^\beta$, be large. As before, β is the normalized data rate in terms of bits per 2D symbol. Then, the first term in (2.29) is a good approximation to the error probability and with large M this becomes

$$P_e \simeq 4\,Q\left[\left(\frac{\mathrm{CFM}(\mathbf{C}_b)}{2}\,\mathrm{SNR}_{M\times M}\right)^{1/2}\right] \tag{2.36}$$

Forney [24] defines the normalized SNR to be

$$\mathrm{SNR}_{\mathrm{norm}} = \mathrm{SNR}_{M\times M}/(2^\beta - 1) \tag{2.37}$$

Substituting (2.20) and (2.37) into (2.36) gives

$$P_e \simeq 4\,Q\left[\left(\frac{3}{2^\beta - 1}\,\mathrm{SNR}_{M\times M}\right)^{1/2}\right] = 4\,Q\left(\sqrt{3\,\mathrm{SNR}_{\mathrm{norm}}}\right) \tag{2.38}$$

When constellation shaping and coding are used, this formula can be used to approximate the symbol error probability except with the normalized SNR scaled by the coding and shaping gains. The coding and shaping gains are discussed in Sections 2.5 and 3.11.

2.2.4 The Continuous Approximation

When the size of a constellation $\mathbf{C}$ is large and all its points lie in a region $\mathbb{R}$, a good approximation for a variety of computations is to assume the constellation points have a continuous uniform probability distribution over $\mathbb{R}$. For example, the continuous approximation to the average energy of the constellation is

$$P(\mathbb{R}) = \int_{\mathbb{R}} \|\mathbf{x}\|^2 \frac{\mathbf{dv}(\mathbf{x})}{V(\mathbb{R})} \tag{2.39}$$

where

$$V(\mathbb{R}) = \int_{\mathbb{R}} \mathbf{dv}(\mathbf{x}) \tag{2.40}$$

is the volume of the region $\mathbb{R}$. The normalized power per two dimensions is defined as

$$\mathbb{P}(\mathbb{R}) = \frac{2}{N} P(\mathbb{R}) \tag{2.41}$$

EXAMPLE 2.4 Continuous Approximation for the Average Power of the N-Cube Grid

An N-cube centered at the origin with side M is the continuous set of points

$$\mathbb{R} = \{(x_1, x_2, \ldots, x_N) \,|\, x_i \in (-M/2, M/2) \text{ for } i = 1, 2, \ldots, N\} \tag{2.42}$$

The volume of the N-cube is $V(\mathbb{R}) = M^N$. Its average power is

$$\begin{aligned} P(\mathbb{R}) &= \frac{1}{M^N} \int_{-M/2}^{M/2} \int_{-M/2}^{M/2} \cdots \int_{-M/2}^{M/2} (x_1^2 + x_2^2 + \cdots + x_N^2)\, dx_1\, dx_2 \cdots dx_N \\ &= N\frac{M^2}{12} = N\frac{2^\beta}{12} \end{aligned} \tag{2.43}$$

and the normalized power per two dimensions is

$$\mathbb{P}(\mathbb{R}) = \frac{2}{N}P(\mathbb{R}) = \frac{M^2}{6} = \frac{2^\beta}{6} = \mathbb{P}_\oplus(\beta) \tag{2.44}$$

For large M these are essentially the same as (2.30) and (2.31) for the N-cube grid. Since $V^{2/N}(\mathbb{R}) = M^2$, the average power can also be written as

$$\mathbb{P}(\mathbb{R}) = \frac{V^{2/N}(\mathbb{R})}{6} \tag{2.45}$$

■

2.3 Constituent 2D Constellations and Constellation Expansion Ratio

A QAM modem transmits a point from an $N = 2L$ dimensional constellation $\mathbf{C}$ by sending a sequence of $L = N/2$ 2D symbols. A *constituent 2D constellation* is the set of 2D points taken on by the particular 2D symbol as the N-dimensional point ranges over all 2^b points in $\mathbf{C}$. If the constituent 2D constellation is the same for each of the L QAM symbols, it will be denoted by $\langle \mathbf{C} \rangle_2$. Because of boundary effects, some of the 2D constituent constellations may contain slightly more or less points than others. In this case, the union of the L 2D constituent constellations will be called $\langle \mathbf{C} \rangle_2$.

When the continuous approximation is used and the N-dimensional points are distributed over the region $\mathbb{R}$, a *constituent 2D region* is defined as the set of 2D points taken on by a particular 2D symbol as the N-dimensional points range over $\mathbb{R}$. If all the constituent 2D regions are the same, they will be denoted by $\langle \mathbb{R} \rangle_2$.

Let $\langle \mathbf{C} \rangle_2^{N/2}$ be the Cartesian product of $\langle \mathbf{C} \rangle_2$ with itself $N/2$ times. It is a set of $|\langle \mathbf{C} \rangle_2|^{N/2}$ N-dimensional points. The original constellation $\mathbf{C}$ is a subset of this Cartesian product. Therefore

$$|\mathbf{C}| \le |\langle \mathbf{C} \rangle_2^{N/2}| = |\langle \mathbf{C} \rangle_2|^{N/2} \tag{2.46}$$

so the number of points in the 2D constituent constellation satisfies the lower bounding inequality

$$|\langle \mathbf{C} \rangle_2| \ge |\mathbf{C}|^{2/N} = 2^{2b/N} = 2^\beta \tag{2.47}$$

This bound holds even if β is not an integer.

The *constellation expansion ratio* is defined to be

$$\mathrm{CER}(\mathbf{C}) = \frac{|\langle \mathbf{C} \rangle_2|}{2^\beta} = \frac{|\langle \mathbf{C} \rangle_2|}{|\mathbf{C}|^{2/N}} \tag{2.48}$$

Usually, one of the design goals for constellations is to keep the constellation expansion ratio as close to 1 as possible.

EXAMPLE 2.5 CER($\mathbf{C}$) for the N-Cube Grid

The N-cube grid $\mathbf{C}_c$ described in Example 2.3 has M^N points. Its constituent 2D constellations are the $M \times M$ square grid $\mathbf{C}_b$ discussed in Example 2.2 which has M^2 points. Therefore, the constellation expansion ratio is

$$\mathrm{CER}(\mathbf{C}_c) = \frac{|\mathbf{C}_b|}{|\mathbf{C}_c|^{2/N}} = \frac{M^2}{(M^N)^{2/N}} = 1 \tag{2.49}$$

■

2.4 Peak-to-Average Power Ratio

Typical communication channels have nonlinearities that limit large signal voltage levels. One cause is the finite power supply voltages for amplifiers. Another is the intentional logarithmic limiting by μ and a-law voiceband codecs called companding. Therefore, a desirable property for a signal constellation is that large constellation points occur with small probability. The *peak-to-average power ratio* (PAR) is a parameter often used to indicate the sensitivity of a constellation to channel nonlinearities. Since QAM modems transmit 2D symbols, we will define PAR in terms of 2D quantities. Let the N-dimensional signal constellation $\mathbf{C}$ have the constituent 2D constellation $\langle \mathbf{C} \rangle_2$ and average power per two dimensions $\mathbb{P}(\mathbf{C})$. Its PAR is defined as

$$\mathrm{PAR}(\mathbf{C}) = \frac{\max_{\mathbf{x} \in \langle \mathbf{C} \rangle_2} \|\mathbf{x}\|^2}{\mathbb{P}(\mathbf{C})} \tag{2.50}$$

2.4.1 PAR for the $M \times M$ Square Grid and N-Cube Grid

According to (2.18) the normalized power for the $M \times M$ square grid is

$$P(\mathbf{C}_b) = \mathbb{P}(\mathbf{C}_b) = \frac{M^2 - 1}{6} d^2$$

The four corner points have the same peak power $P_{max} = 2[(M-1)d/2]^2$. Therefore,

$$\mathrm{PAR}(\mathbf{C}_b) = \frac{(M-1)^2 d^2/2}{(M^2-1)d^2/6} = 3\frac{M-1}{M+1} \tag{2.51}$$

The N-cube grid is the Cartesian product of $N/2$ $M \times M$ square grids and so its PAR is the same as that of the $M \times M$ square grid. The PAR approaches 3 as M becomes large.

The continuous approximation is quite accurate for large M. Suppose the constellation points are uniformly distributed over a square $\mathbb{R}$ with sides extending from $-M/2$ to $M/2$. The four corner points are the ones with peak power which is $2(M/2)^2$. According to (2.44) the normalized power is $M^2/6$ so the PAR is

$$\text{PAR}(\mathbb{R}) = \frac{M^2/2}{M^2/6} = 3 \tag{2.52}$$

which is exactly the limit of the PAR for the $M \times M$ square grid as M becomes infinite.

2.4.2 PAR for a Circle

The continuous approximation will be used to estimate the PAR for a constellation of large size with its points confined to a region $\mathbb{R}$ that is the interior of a circle of radius R centered on the origin. Let the points be continuously and uniformly distributed over this circle. The two dimensional probability density function for the points $\mathbf{x} = (x, y)$ is

$$f(x,y) = \begin{cases} \frac{1}{\pi R^2} & \text{for } x^2 + y^2 \le R^2 \\ 0 & \text{elsewhere} \end{cases} \tag{2.53}$$

Clearly, the peak power is R^2. The average power is

$$P(\mathbb{R}) = \frac{1}{\pi R^2} \int\int_{\mathbb{R}} x^2 + y^2 \, dx \, dy \tag{2.54}$$

Switching to polar coordinates by letting $r^2 = x^2 + y^2$ and $dx\,dy = r\,d\theta\,dr$ gives

$$P(\mathbb{R}) = \frac{1}{\pi R^2} \int_0^R \int_0^{2\pi} r^3 \, d\theta \, dr = \frac{R^2}{2} \tag{2.55}$$

Therefore

$$\text{PAR}(\mathbb{R}) = \frac{R^2}{R^2/2} = 2 \tag{2.56}$$

It is interesting to observe that even though the constellation points are uniformly distributed in two dimensions, the one-dimensional marginal pdf's are not uniform. The pdf's in the x and y directions both have the form

$$f(x) = \begin{cases} \frac{2}{\pi R^2}\sqrt{R^2 - x^2} & \text{for } |x| \le R \\ 0 & \text{elsewhere} \end{cases} \tag{2.57}$$

which has the shape of a semi-circle over the x axis with radius R.

A circle is the optimum shape in two dimensions in the sense that it has the smallest average power of any figure with the same area and center of gravity at the origin. As an example, a square and a circle with the same area are shown in Figure 2.2. A circle with radius R and a square with side M have the same area if

$$R = \frac{2}{\sqrt{\pi}} \frac{M}{2} = 1.1284 \frac{M}{2} \tag{2.58}$$

so the diameter of the circle is larger than the side of the square. A heuristic proof of the optimality property can be made using this figure. Each of the four regions of the square that extend outside the circle are matched by four regions of the circle that extend outside the square and have the same areas. If points in a region of the square outside the circle are moved into a region of the circle outside the square, the squared magnitudes of the points is reduced and the average power becomes less. This same argument can be used for any figure that extends outside the circle.

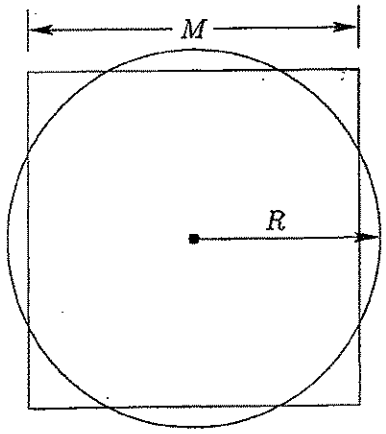


Figure 2.2. A Square Superimposed on a Circle with the Same Area

The points for the circle take on a larger range of values than for the square when projected onto the one-dimensional x or y axes. This is an elementary example of constellation expansion resulting from shaping and will be discussed in more detail later.

EXHIBIT E
Part 2

2.4.3 PAR for the N-Sphere

An N-sphere of radius R is the set of N-tuples $\mathbf{x} = (x_1, x_2, \ldots, x_N)$ such that

$$\|\mathbf{x}\|^2 = x_1^2 + x_2^2 + \cdots + x_N^2 = R^2$$

A circle is the special case for $N = 2$. The volume of an N-sphere is [68]

$$V(R) = \begin{cases} \dfrac{2^N \pi^{(N-1)/2} \left(\dfrac{N-1}{2}\right)!}{N!} R^N & \text{for } N \text{ odd} \\ \dfrac{\pi^{N/2}}{\left(\dfrac{N}{2}\right)!} R^N & \text{for } N \text{ even} \end{cases} \tag{2.59}$$

Let $\mathbb{R}$ be the N-dimensional region inside and including the sphere. The average power for this region is

$$\begin{aligned} P(\mathbb{R}) &= \frac{1}{V(R)} \int_{\mathbb{R}} \left(x_1^2 + x_2^2 + \cdots + x_N^2\right) dx_1\, dx_2 \cdots dx_N \\ &= \frac{1}{V(R)} \int_0^R r^2\, dV(r) \end{aligned} \tag{2.60}$$

Differentiating $V(r)$ with respect to r, it follows that

$$\frac{dV(r)}{V(R)} = \frac{N r^{N-1}}{R^N}\, dr \quad \text{for } N \text{ even or odd} \tag{2.61}$$

So

$$P(\mathbb{R}) = \int_0^R \frac{N}{R^N} r^{N+1}\, dr = \frac{N}{N+2} R^2 \tag{2.62}$$

As a result of the symmetry of the N-sphere, the points in each constituent 2D region have the same distribution. Also, the average power of $\mathbb{R}$ is the sum of the powers of its $N/2$ constituent 2D regions. Therefore, the average power of a constituent 2D region must be

$$\mathbb{P}(\mathbb{R}) = \frac{2}{N} P(\mathbb{R}) = \frac{2}{N+2} R^2 \tag{2.63}$$

The peak power in a constituent 2D constellation is R^2 so the PAR is

$$\text{PAR}(\mathbb{R}) = \frac{R^2}{\dfrac{2}{N+2} R^2} = \frac{N}{2} + 1 \tag{2.64}$$

Observe that the PAR becomes infinite as N becomes infinite unlike for the N-cube whose PAR is 3 for all N.

The PAR formula just derived was for the constituent 2D region. The picture is different if the PAR is referenced to N dimensions. The peak power of points in the N-sphere is still R^2. However, the average power is $(N/2)\mathbb{P}(\mathbb{R})$, so the PAR in N dimensions is

$$\text{PAR}_N(\mathbb{R}) = \frac{\text{PAR}(\mathbb{R})}{(N/2)} = \frac{N+2}{N} \tag{2.65}$$

This approaches 1 as N becomes large, so the average power becomes equal to the peak power. This demonstrates the curious fact that almost all the volume of an N-sphere is at its boundary for large N.

It is interesting to compare the radius of a $2N$-sphere and half side length of a $2N$-cube with the same volume. From (2.59), the volume of a $2N$ sphere is $(\pi R^2)^N/N!$ and the volume of a $2N$-cube is M^{2N}. Equating these volumes and solving for R gives

$$R = \frac{M}{2} \frac{2(N!)^{\frac{1}{2N}}}{\sqrt{\pi}} = \frac{M}{2} k(N) \tag{2.66}$$

Taking the logarithm of $k(N)$ and using the integral test, it is found that $k(N)$ diverges as N becomes infinite. Therefore, we find the surprising fact that the radius of the $2N$-sphere becomes infinite relative to the half side length of the $2N$-cube of the same volume as N becomes infinite. Using Sterling's approximation, $N! \simeq \sqrt{2\pi N}(N/e)^N$ for large N, (2.66) becomes

$$R \simeq \frac{M}{2} \sqrt{\frac{4N}{\pi e}} \tag{2.67}$$

The optimality property of the circle generalizes to the N-sphere. The N-sphere has the smallest average power of any N-dimensional figure with the same volume.

2.5 Representing CFM(C) in Terms of Coding Gain and Shaping Gain

In this section we will see that for constellations of large size, the constellation figure of merit CFM(C) introduced in Section 2.2.2 can be well approximated by the product of three factors: (1) the baseline constellation figure of merit $\text{CFM}_{\oplus}(\beta)$ defined by (2.33), (2) the fundamental coding gain $\gamma_c(\Lambda)$ discussed in Section 1.7, and (3) the shaping gain $\gamma_s(\mathbb{R})$ which will be defined shortly. The coding gain is a measure of the density of the lattice used for the constellation relative to the baseline N-cube grid. The shaping gain is a measure of the reduction in power obtained by confining the constellation points to the region

$\mathbb{R}$ rather than an N-cube. These gains are essentially uncoupled. We will see that the shaping gain can be at most $\pi e/6 = 1.53$ dB.

We will assume the constellation is a lattice code $\mathbf{C}(\Lambda, \mathbb{R})$ with a large size $|\mathbf{C}| = 2^b$ or normalized rate $\beta = 2b/N$ bits per 2D. We have already observed in (2.35) that the constellation figure of merit gain relative to the baseline N-cube grid is

$$\frac{\mathrm{CFM}(\mathbf{C})}{\mathrm{CFM}_{\oplus}(\beta)} = \frac{2^{\beta} d^2_{\min}(\mathbf{C})}{6\mathbb{P}(\mathbf{C})} \tag{2.68}$$

The minimum squared distances for $\mathbf{C}$ and Λ are the same, that is, $d^2_{\min}(\mathbf{C}) = d^2_{\min}(\Lambda)$. The fundamental volume $V(\Lambda)$ is associated with each lattice point and the constellation is confined to a region $\mathbb{R}$ with volume $V(\mathbb{R})$. Therefore, to a good approximation, the constellation size is

$$|\mathbf{C}| = 2^{\beta N/2} \simeq \frac{V(\mathbb{R})}{V(\Lambda)} \tag{2.69}$$

so

$$2^{\beta} \simeq \left[\frac{V(\mathbb{R})}{V(\Lambda)}\right]^{2/N} \tag{2.70}$$

and the baseline figure of merit is approximately

$$\mathrm{CFM}_{\oplus}(\beta) = \frac{6}{2^{\beta}} \simeq 6\left[\frac{V(\Lambda)}{V(\mathbb{R})}\right]^{2/N} \tag{2.71}$$

According to the continuous approximation, the normalized constellation power $\mathbb{P}(\mathbf{C})$ can be approximated by the normalized power $\mathbb{P}(\mathbb{R})$ of the region $\mathbb{R}$. Substituting these results into (2.68) gives the approximation

$$\frac{\mathrm{CFM}(\mathbf{C})}{\mathrm{CFM}_{\oplus}(\beta)} \simeq \frac{V^{2/N}(\mathbb{R})\, d^2_{\min}(\Lambda)}{V^{2/N}(\Lambda)\, 6\,\mathbb{P}(\mathbb{R})} \tag{2.72}$$

$$= \frac{d^2_{\min}(\Lambda)}{V^{2/N}(\Lambda)} \frac{V^{2/N}(\mathbb{R})}{6\,\mathbb{P}(\mathbb{R})} = \gamma_c(\Lambda)\,\gamma_s(\mathbb{R}) \tag{2.73}$$

The first factor

$$\gamma_c(\Lambda) = \frac{d^2_{\min}(\Lambda)}{V^{2/N}(\Lambda)} \tag{2.74}$$

is the fundamental coding gain of the lattice Λ already discussed in Section 1.7. The second factor

$$\gamma_s(\mathbb{R}) = \frac{V^{2/N}(\mathbb{R})}{6\,\mathbb{P}(\mathbb{R})} \tag{2.75}$$

is called the *shaping gain* of the region $\mathbb{R}$.

2.5.1 Why $\gamma_c(\Lambda)$ is Called the Fundamental Coding Gain

Justification for calling $\gamma_c(\Lambda)$ the fundamental coding gain will now be given. The symbol error probability for a constellation is very strongly dependent on its minimum squared distance at large SNR. Therefore, it is reasonable to compare two constellations when they have been scaled to have the same $d_{\min}$. Suppose an N-dimensional constellation $\mathbf{C}_c$ with $|\mathbf{C}_c| = M = 2^{\beta N/2}$ points is to be designed by selecting points from a lattice $\Lambda' = \Lambda/d_{\min}(\Lambda)$. This lattice has the minimum squared distance $d_{\min}(\Lambda') = 1$. The constellation can be formed by selecting the M points of Λ' closest to the origin that lie in an N-sphere centered on the origin with a radius R_c sufficiently large. For large M, the volume of this sphere is approximately $V_c = MV(\Lambda')$ where

$$V(\Lambda') = V[\Lambda/d_{\min}(\Lambda)] = V(\Lambda)/d^N_{\min}(\Lambda) \tag{2.76}$$

is the fundamental volume for the lattice. As the baseline "uncoded" constellation $\mathbf{C}_u$, let the M points be selected from the lattice $\mathbf{Z}^N$. The baseline constellation has $d_{\min} = 1$ also. The volume of the sphere for the baseline constellation is approximately $V_u = MV(\mathbf{Z}^N) = M$. Let its radius be R_u. The subscript c is intended to suggest "coded" and the subscript u "uncoded." According to (2.59), the volume of an N-sphere is proportional to the Nth power of its radius, so

$$\frac{V_u}{V_c} = \left(\frac{R_u}{R_c}\right)^N = \frac{MV(\mathbf{Z}^N)}{MV(\Lambda')} = \frac{d^N_{\min}(\Lambda)}{V(\Lambda)} \tag{2.77}$$

Therefore,

$$\frac{R_u^2}{R_c^2} = \frac{d^2_{\min}(\Lambda)}{V^{2/N}(\Lambda)} \tag{2.78}$$

Let $P(\mathbf{C}_c)$ be the average power of the constellation $\mathbf{C}_c$ and $\mathbb{P}(\mathbf{C}_c) = \frac{2}{N}P(\mathbf{C}_c)$ be its power normalized to two dimensions, and similarly for the baseline constellation $\mathbf{C}_u$. According to (2.62), the average power of an N-sphere is proportional to the square of its radius, so the ratio of the average powers of the uncoded and coded constellations is closely approximated by

$$\frac{P(\mathbf{C}_u)}{P(\mathbf{C}_c)} = \frac{\mathbb{P}(\mathbf{C}_u)}{\mathbb{P}(\mathbf{C}_c)} \simeq \frac{d^2_{\min}(\Lambda)}{V^{2/N}(\Lambda)} = \gamma_c(\Lambda) \tag{2.79}$$

One interpretation of this result is that the coded constellation requires less power by the factor of the coding gain than the baseline constellation to achieve essentially the same error probability. Another interpretation is that the coded constellation can be scaled up by the factor $\gamma_c^{1/2}$ to have the same power as the baseline constellation but with a large $d_{\min}$ and smaller error probability.

2.5.2 Shaping Gain Properties and Examples

The shaping gain of a region $\mathbb{R}$ is a measure of the power efficiency improvement gained by using a constellation uniformly distributed over a region $\mathbb{R}$ that is more spherical than the baseline N-cube. The numerator and denominator both have units of squared length, so the shaping gain is a dimensionless parameter.

As a reference point, the volume of an N-cube centered at the origin with a side of length M is $V(\mathbb{R}) = M^N$ and its normalized average power is $\mathbb{P}(\mathbb{R}) = M^2/6$ according to (2.44). Therefore, the shaping gain of the baseline N-cube is

$$\gamma_s(\mathbb{R}) = \frac{V^{2/N}(\mathbb{R})}{6\,\mathbb{P}(\mathbb{R})} = \frac{(M^N)^{2/N}}{6(M^2/6)} = 1 \tag{2.80}$$

Consider an N-dimensional region $\mathbb{R}$ with volume $V(\mathbb{R})$. An N-cube with the same volume must have the squared side $M^2 = V^{2/N}(\mathbb{R})$. The normalized average power for this N-cube is

$$\mathbb{P}_{\oplus}(\mathbb{R}) = \frac{V^{2/N}(\mathbb{R})}{6} \tag{2.81}$$

Therefore, the shaping gain can also be written as

$$\gamma_s(\mathbb{R}) = \frac{\mathbb{P}_{\oplus}(\mathbb{R})}{\mathbb{P}(\mathbb{R})} \tag{2.82}$$

In words, the shaping gain is the ratio of the normalized average power of an N-cube with the same volume as $\mathbb{R}$ to the normalized average power of $\mathbb{R}$.

Let $\gamma_s^{(u)}(\langle\mathbb{R}\rangle_2)$ be the shaping gain of the constituent 2D region $\langle\mathbb{R}\rangle_2$ when the points are uniformly distributed over this region. The *biasing gain* is defined as

$$\mathrm{B}(\mathbb{R}) = \frac{\gamma_s(\mathbb{R})}{\gamma_s^{(u)}(\langle\mathbb{R}\rangle_2)} \tag{2.83}$$

One way of achieving biasing gain or, equivalently, shaping gain is to choose the constellation points uniformly distributed over an N-dimensional region, such as an N-sphere, selected to induce a desirable nonuniform distribution over the constituent 2D region. Calderbank and Ozarow [8] show how to achieve biasing gain by directly setting the distribution of points in the 2D constellation. Actually, they begin with a generalization to N-dimensional constellations. First, an N-dimensional region $\mathbb{R}_0$ is chosen. Then the nested sequence of T scaled regions $\mathbb{R}_i = (i+1)^{1/N}\mathbb{R}_0$ for $i = 0, \ldots, T-1$ is formed. The volume of the ith region is related to $V(\mathbb{R}_0)$ by $V(\mathbb{R}_i) = (i+1)V(\mathbb{R}_0)$. Constellation points are selected from the largest region $\mathbb{R}_{T-1} = T^{1/N}\mathbb{R}_0$. This region is subdivided into T shells of equal volume consisting of $\mathbb{S}_0 = \mathbb{R}_0$ and

$\mathbb{S}_i = \mathbb{R}_i - \mathbb{R}_{i-1} = \mathbb{R}_i \cap \overline{\mathbb{R}}_{i-1}$ for $i = 1, \ldots, T-1$. The volume of each shell is $V(\mathbb{R}_0)$. If the constellation points are selected from the lattice Λ, the points in each shell are $\Lambda \cap \mathbb{S}_i$ and in the continuous approximation each shell has the same number of points. Calderbank and Ozarow propose using all points in a shell with the same frequency. Then shells are selected with frequencies $f_0, f_1, \ldots, f_{T-1}$ by a shaping code. The shell frequencies can be selected to maximize the biasing gain. For QAM modulation, we would let $N = 2$ and probably chose $\mathbb{R}_0$ as a circle so the shells become circular rings. This approach allows constellation points to be addressed much more easily than shaping by selecting points in an N-sphere or Voronoi regions of multidimensional lattices [26]. For comparable shaping gain and constellation expansion ratio, the PAR for shaping by shells is superior.

The shaping gain is related to the *normalized second moment* of the region $\mathbb{R}$ which is defined as

$$G(\mathbb{R}) = \frac{\mathbb{P}(\mathbb{R})}{2V^{2/N}(\mathbb{R})} = \int_{\mathbf{x}\in\mathbb{R}} \|\mathbf{x}\|^2 \frac{\mathbf{dv}(\mathbf{x})}{V(\mathbb{R})} \times \frac{1}{NV^{2/N}(\mathbb{R})} \tag{2.84}$$

For reference, the normalized second moment of the N-cube is

$$G_{\oplus} = \frac{M^2/6}{2(M^N)^{2/N}} = \frac{1}{12} \tag{2.85}$$

Therefore, the shaping gain is related to the normalized second moment by the equation

$$\gamma_s(\mathbb{R}) = \frac{G_{\oplus}}{G(\mathbb{R})} = \frac{1/12}{G(\mathbb{R})} \tag{2.86}$$

It is a measure of the reduction in normalized second moment achieved by using the region $\mathbb{R}$ instead of an N-cube.

Some additional properties of the shaping gain are:

(a) It is invariant to scaling, that is, $\gamma_s(c\mathbb{R}) = \gamma_s(\mathbb{R})$ where c is a scalar. This result is easily proved by substituting the formulas $V(c\mathbb{R}) = c^N V(\mathbb{R})$ and $\mathbb{P}(c\mathbb{R}) = c^2\mathbb{P}(\mathbb{R})$ into the shaping gain definition (2.75).

(b) It is invariant to orthogonal transformations of $\mathbb{R}$. Let $\mathbf{A}$ be an orthogonal matrix with $\mathbf{A}\mathbf{A}^t = c^2\mathbf{I}$ and $|\det \mathbf{A}| = |c|^N$ as discussed on page 23 where a similar property is demonstrated for the coding gain $\gamma_c(\Lambda)$. Let $\tilde{\mathbb{R}} = \{\tilde{\mathbf{r}} = \mathbf{r}\mathbf{A} \,|\, \mathbf{r} \in \mathbb{R}\}$ be the transformed region. Then $\gamma_s(\tilde{\mathbb{R}}) = \gamma_s(\mathbb{R})$.

To see why this is true, first observe that the Jacobian of the transformation $\tilde{\mathbf{r}} = \mathbf{r}\mathbf{A}$ is $J = \det \mathbf{A}$ [38]. The volume dV of an incremental region in $\mathbb{R}$ and the volume $d\tilde{V}$ of the image of this region in $\tilde{\mathbb{R}}$ are related by $d\tilde{V} =$

$|\det \mathbf{A}|\, dV = |c|^N dV$. Therefore, $V(\tilde{\mathbb{R}}) = |c|^N V(\mathbb{R})$ and $d\tilde{V}/V(\tilde{\mathbb{R}}) = dV/V(\mathbb{R})$. The average power of the transformed region is

$$\begin{aligned} P(\tilde{\mathbb{R}}) &= \int_{\tilde{\mathbb{R}}} \|\tilde{\mathbf{r}}\|^2 \frac{d\tilde{V}}{V(\tilde{\mathbb{R}})} = \int_{\mathbb{R}} \mathbf{r}\mathbf{A}\mathbf{A}^t\mathbf{r}^t \frac{dV}{V(\mathbb{R})} \\ &= c^2 \int_{\mathbb{R}} \mathbf{r}\mathbf{r}^t \frac{dV}{V(\mathbb{R})} = c^2 P(\mathbb{R}) \end{aligned} \tag{2.87}$$

and the power normalized to two dimensions is

$$\mathbb{P}(\tilde{\mathbb{R}}) = (2/N) P(\tilde{\mathbb{R}}) = c^2 \mathbb{P}(\mathbb{R}) \tag{2.88}$$

Putting these results together gives

$$\gamma_s(\tilde{\mathbb{R}}) = \frac{V^{2/N}(\tilde{\mathbb{R}})}{6\mathbb{P}(\tilde{\mathbb{R}})} = \frac{c^2 V^{2/N}(\mathbb{R})}{6c^2\mathbb{P}(\mathbb{R})} = \gamma_s(\mathbb{R}) \tag{2.89}$$

(c) The Cartesian product $\mathbb{R}^M$ has the same shaping gain as $\mathbb{R}$. This can be shown by observing that the average power for this MN-dimensional region is $P(\mathbb{R}^M) = MP(\mathbb{R})$ so its normalized average power is $\mathbb{P}(\mathbb{R}^M) = 2P(\mathbb{R}^M)/(MN) = \mathbb{P}(\mathbb{R})$. Also, $V(\mathbb{R}^M) = V^M(\mathbb{R})$. Therefore,

$$\gamma_s(\mathbb{R}^M) = \frac{V^{2/(MN)}(\mathbb{R}^M)}{6\mathbb{P}(\mathbb{R}^M)} = \frac{V^{2/N}(\mathbb{R})}{6\mathbb{P}(\mathbb{R})} = \gamma_s(\mathbb{R}) \tag{2.90}$$

EXAMPLE 2.6 Shaping Gain of a Circle

Figure 2.2 shows a square superimposed on a circle of the same area. The area of the circle is $V = \pi R^2$ and its average power is $\mathbb{P} = R^2/2$ according to (2.55). The normalized second moment for the circle is

$$G(\mathbb{R}) = \frac{\mathbb{P}}{NV^{2/N}} = \frac{R^2/2}{2\pi R^2} = \frac{1}{4\pi} \tag{2.91}$$

and its shaping gain is

$$\gamma_s(\mathbb{R}) = \frac{1/12}{G(\mathbb{R})} = \frac{\pi}{3} = 0.20028 \text{ dB} \tag{2.92}$$

■

EXAMPLE 2.7 Shaping Gain for an N-Sphere

Substituting (2.59) for the volume of an N-sphere and (2.62) for its average power into the definition of the normalized second moment (2.84), gives

$$G(\mathbb{R}) = \begin{cases} \dfrac{(N!)^{2/N}}{(N+2)4\pi^{(N-1)/N}\left[\left(\frac{N-1}{2}\right)!\right]^{2/N}} & \text{for } N \text{ odd} \\ \dfrac{\left[\left(\frac{N}{2}\right)!\right]^{2/N}}{(N+2)\pi} & \text{for } N \text{ even} \end{cases} \tag{2.93}$$

The shaping gain of the N-sphere is

$$\gamma_s(\mathbb{R}) = \frac{1}{12G(\mathbb{R})} = \begin{cases} \dfrac{(N+2)\pi^{(N-1)/N}\left[\left(\frac{N-1}{2}\right)!\right]^{2/N}}{3(N!)^{2/N}} & \text{for } N \text{ odd} \\ \dfrac{\pi\left(\frac{N}{2}+1\right)}{6\left[\left(\frac{N}{2}\right)!\right]^{2/N}} & \text{for } N \text{ even} \end{cases} \tag{2.94}$$

The shaping gain is plotted *vs.* N in Figure 2.3.

It was pointed out in Section 2.4.3 that the N-sphere has the smallest average power of any N-dimensional figure with the same volume. Therefore, the N-sphere is optimal in the sense that it has the largest possible shaping gain of any N-dimensional region. ■

2.5.3 The Ultimate Achievable Shaping Gain and Corresponding 2D Distribution

The shaping gain of an N-sphere monotonically increases with N. The ultimate shaping gain can be found by letting $N = 2L$ and using Sterling's approximation $L! \simeq \sqrt{2\pi L}(L/e)^L$ in the part of (2.94) for even N with the result

$$\begin{aligned} \lim_{N\to\infty} \gamma_s(\mathbb{R}) &= \lim_{L\to\infty} \frac{\pi(L+1)}{6(L!)^{1/L}} = \lim_{L\to\infty} \frac{\pi(L+1)}{6\left[\sqrt{2\pi L}(L/e)^L\right]^{1/L}} \\ &= \frac{\pi e}{6} = 1.423289 = 1.53293 \text{ dB} \end{aligned} \tag{2.95}$$

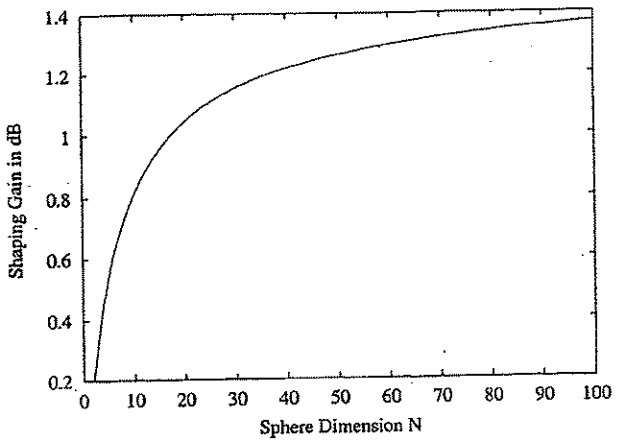


Figure 2.3. Shaping Gain $\gamma_s(\mathbb{R})$ for an N-sphere vs. N

From Figure 2.3 we see that the shaping gain increases rapidly with N at first but then increases quite slowly. In fact, computations show that the shaping gain does not reach 1.530 dB until N increases to about 12800.

The Calderbank and Ozarow [8] method of shaping by shells briefly described on page 44 can also achieve this ultimate shaping gain. They show that by selecting T rings with optimum probabilities in two dimensions and then taking the limit as T becomes infinite, the ultimate shaping gain of $\pi e/6$ is reached when the basic two-dimensional region $\mathbb{R}_0$ is a circle.

2.5.3.1 Convergence of the Distribution for the Constituent 2D Region to a Circular 2D Gaussian Distribution

Let $\mathbf{X} = (X_1, X_2, \ldots, X_N)$ be a random vector uniformly distributed over the region bounded by an N-sphere of radius R centered at the origin and let $V_N(r)$ be the volume of an N-sphere of radius r. Then the probability density function (pdf) for $\mathbf{X}$ is

$$f(x_1, x_2, \ldots, x_N) = \begin{cases} \frac{1}{V_N(R)} & \text{for } x_1^2 + x_2^2 + \ldots + x_N^2 \le R^2 \\ 0 & \text{elsewhere} \end{cases} \tag{2.96}$$

The 2D marginal pdf for X_1 and X_2 for $x_1^2 + x_2^2 \le R^2$ is

$$\begin{aligned} f(x_1, x_2) &= \int\cdots\int_{x_1^2+\cdots+x_N^2 \le R^2} f(x_1, x_2, x_3, \ldots, x_N)\, dx_3 \cdots dx_N \\ &= \frac{1}{V_N(R)} \int\cdots\int_{x_3^2+\cdots+x_N^2 \le R^2-x_1^2-x_2^2} dx_3 \cdots dx_N \\ &= \frac{V_{N-2}\left(\sqrt{R^2 - x_1^2 - x_2^2}\right)}{V_N(R)} \end{aligned} \tag{2.97}$$

When N is even, it follows from (2.59) that this pdf is

$$f(x_1, x_2) = \begin{cases} \frac{N}{2\pi R^2}\left(1 - \frac{x_1^2 + x_2^2}{R^2}\right)^{\frac{N-2}{2}} & \text{for } x_1^2 + x_2^2 \le R^2 \\ 0 & \text{elsewhere} \end{cases} \tag{2.98}$$

Notice that for any nonzero point (x_1, x_2) strictly inside the circle of radius R, this joint pdf converges to zero as N becomes infinite so the pdf becomes highly concentrated around the origin. When the ratio $(x_1^2 + x_2^2)/R^2$ is small, the approximation $e^{-x} \simeq 1 - x$ for $|x| \ll 1$ can be used in (2.98) to find that for large N

$$f(x_1, x_2) \simeq \frac{1}{2\pi R^2/N} e^{-\frac{x_1^2+x_2^2}{2R^2/N}} \tag{2.99}$$

This is the joint pdf for two independent Gaussian random variables X_1 and X_2 each with zero mean and variance $\sigma^2 = R^2/N$. Since the 2D random variable (X_1, X_2) is confined to a circle of radius R, its peak power is R^2. Its average power is $2\sigma^2$ so the PAR is $R^2/(2R^2/N) = N/2$ which agrees with (2.64) for large N.

2.6 Coding and Shaping Factors of the Constellation Expansion Ratio

Once again in this section, we will assume the signal constellation is an N-dimensional lattice code $\mathbf{C}(\mathbf{\Lambda}, \mathbb{R})$ with large size so the continuous approximation is accurate. The constituent 2D constellations were defined in Section 2.3 as the projections of $\mathbf{C}$ onto selected pairs of coordinates. More precisely, let $\boldsymbol{\lambda}$ be an N-component vector in $\mathbf{\Lambda}$, and i and j be two distinct coordinate indices. Then the corresponding constituent 2D constellation is

$$\langle \mathbf{C} \rangle_{2(i,j)} = \{(\lambda_i, \lambda_j) \,|\, \boldsymbol{\lambda} \in \mathbf{\Lambda} \cap \mathbb{R}\} \tag{2.100}$$

The *constituent 2D lattice* $\langle \mathbf{\Lambda} \rangle_{2(i,j)}$ is the infinite discrete set of points

$$\langle \mathbf{\Lambda} \rangle_{2(i,j)} = \{(\lambda_i, \lambda_j) \,|\, \boldsymbol{\lambda} \in \mathbf{\Lambda}\} \tag{2.101}$$

The set $\langle \mathbf{\Lambda} \rangle_{2(i,j)}$ is indeed a lattice. Let $\mathbf{u}$ and $\mathbf{v}$ be points in the N-dimensional lattice $\mathbf{\Lambda}$ so that $\mathbf{u} + \mathbf{v} \in \mathbf{\Lambda}$ and the projection of this sum onto the (i, j) pair

of coordinates is (u_i+v_i, u_j+v_j). Clearly (u_i, u_j) and (v_i, v_j) both belong to $\langle\Lambda\rangle_{2(i,j)}$. But their sum (u_i+v_i, u_j+v_j) also belongs to $\langle\Lambda\rangle_{2(i,j)}$ so it must be a lattice.

Similarly, the *constituent 2D regions* are defined as the projections of $\mathbb{R}$ onto selected pairs of coordinates. More precisely, let i and j be two distinct coordinate indices for the N-tuple $\mathbf{r}$. Then the corresponding constituent 2D region is

$$\langle\mathbb{R}\rangle_{2(i,j)} = \{(r_i, r_j) \,|\, \mathbf{r} \in \mathbb{R}\} \tag{2.102}$$

If $\langle\mathbf{C}\rangle_{2(i,j)}$, $\langle\Lambda\rangle_{2(i,j)}$ and $\langle\mathbb{R}\rangle_{2(i,j)}$ do not depend on the choice of coordinate pairs (i,j), then $\mathbf{C}$, Λ and $\mathbb{R}$ are called *2D symmetric*. In this case, the (i,j) subscripts can be dropped with $\langle\mathbf{C}\rangle_2$ called the constituent 2D constellation, $\langle\Lambda\rangle_2$ the constituent 2D lattice, and $\langle\mathbb{R}\rangle_2$ the constituent 2D region.

Suppose the 2D symmetric condition holds. An N-dimensional lattice $\Gamma = \langle\Lambda\rangle_2^{N/2}$ which is the $N/2$-fold Cartesian product of $\langle\Lambda\rangle_2$ with itself can be formed. If $\mathbf{G}_2$ is the 2×2 generator matrix for $\langle\Lambda\rangle_2$, the generator matrix for Γ is the $N\times N$ block diagonal matrix $\mathbf{G}$ with $\mathbf{G}_2$ down its diagonal. According to (1.40) the fundamental volume for Γ is

$$V(\Gamma) = |\det \mathbf{G}| = |\det \mathbf{G}_2|^{N/2} = V^{N/2}(\langle\Lambda\rangle_2) \tag{2.103}$$

where $V(\langle\Lambda\rangle_2)$ is the fundamental volume for $\langle\Lambda\rangle_2$. The lattice Λ is a sublattice of Γ since it is the concatenation of $N/2$ two-tuples each belonging to $\langle\Lambda\rangle_2$. However, not all sequences of two-tuples are allowed, so Γ is denser than Λ. The *normalized redundancy* of a binary lattice Λ is defined as

$$\rho(\Lambda) \triangleq \frac{2}{N}\log_2 |\langle\Lambda\rangle_2^{N/2}/\Lambda| \tag{2.104}$$

$$= \frac{2}{N}\log_2 \frac{V(\Lambda)}{V(\langle\Lambda\rangle_2^{N/2})} = \frac{2}{N}\log_2 \frac{V(\Lambda)}{V^{N/2}(\langle\Lambda\rangle_2)}$$

$$= \log_2 \frac{V^{2/N}(\Lambda)}{V(\langle\Lambda\rangle_2)} \tag{2.105}$$

EXAMPLE 2.8

Let $\Lambda = \mathbf{D}_4$. Then $N/2 = 2$, $\langle\Lambda\rangle_2 = \mathbf{Z}^2$ and $\langle\Lambda\rangle_2^2 = \mathbf{Z}^4$. The normalized redundancy is $\rho(\mathbf{D}_4) = (1/2)\log_2 |\mathbf{Z}^4/\mathbf{D}_4| = 1/2$. ∎

An approximate formula for the constellation expansion ratio will now be derived. Based on the continuous approximation, the size of $\mathbf{C}(\Lambda,\mathbb{R})$ is close to $|\mathbf{C}| \simeq V(\langle\mathbb{R}\rangle)/V(\Lambda)$. Similarly, the size of the constituent 2D constellation is approximately $|\langle\mathbf{C}\rangle_2| \simeq V(\langle\mathbb{R}\rangle_2)/V(\langle\Lambda\rangle_2)$. Therefore, the CER is

approximately

$$\mathrm{CER}(\mathbf{C}) = \frac{|\langle\mathbf{C}\rangle_2|}{|\mathbf{C}|^{2/N}} \simeq \frac{V^{2/N}(\Lambda)}{V(\langle\Lambda\rangle_2)} \times \frac{V(\langle\mathbb{R}\rangle_2)}{V^{2/N}(\langle\mathbb{R}\rangle)}$$

$$= \mathrm{CER}_c(\Lambda) \times \mathrm{CER}_s(\mathbb{R}) \tag{2.106}$$

The first factor

$$\mathrm{CER}_c(\Lambda) = \frac{V^{2/N}(\Lambda)}{V(\langle\Lambda\rangle_2)} \tag{2.107}$$

is called the *coding constellation expansion ratio* of Λ. For the N-dimensional integer lattice $\mathbf{Z}^N$, this factor is $\mathrm{CER}_c(\mathbf{Z}^N) = 1$. The second factor

$$\mathrm{CER}_s(\mathbb{R}) = \frac{V(\langle\mathbb{R}\rangle_2)}{V^{2/N}(\langle\mathbb{R}\rangle)} \tag{2.108}$$

is called the *shaping constellation expansion ratio* of $\mathbb{R}$.

EXAMPLE 2.9 $\mathrm{CER}_s(\mathbb{R})$ for an N-Sphere

The constituent 2D constellations $\langle\mathbb{R}\rangle_2$ for an N-sphere with radius R are circles of radius R. This can be seen by observing that points contained in the N-sphere have the form

$$\mathbf{x} = (x_1, x_2, \ldots, x_N) \text{ with } \sum_{i=1}^{N} x_i^2 \le R^2 \tag{2.109}$$

so N-tuples of the form

$$(x_1, x_2, 0, \ldots, 0) \text{ with } x_1^2 + x_2^2 \le R^2$$

(as well as others) are in the sphere and these points are in a circle of radius R. The point $(x_1 = 0, x_2 = 0)$ at the center of the circle occurs the most ways because the remaining coordinates just have to chosen to satisfy

$$\sum_{i=3}^{N} x_i^2 \le R^2$$

while the points with $x_1^2 + x_2^2 = R^2$ on the boundary occur least often because the only choice for each of the remaining coordinates is zero. The two-dimensional probability density function for the points is similar to a truncated two-dimensional circular Gaussian density function when the N-dimensional points are chosen with equal likelihood in $\mathbb{R}$.

The area of the constituent 2D region is $V(\langle\mathbb{R}\rangle_2) = \pi R^2$. The volume of the N-sphere with N even is according to (2.59)

$$V(\mathbb{R}) = \frac{(\pi R^2)^{N/2}}{(N/2)!} \tag{2.110}$$

Solving this last equation for πR^2 gives

$$V(\langle\mathbb{R}\rangle_2) = \pi R^2 = [(N/2)!]^{2/N} V^{2/N}(\mathbb{R}) \tag{2.111}$$

so

$$\mathrm{CER}_s(\mathbb{R}) = \frac{V(\langle\mathbb{R}\rangle_2)}{V^{2/N}(\mathbb{R})} = [(N/2)!]^{2/N} \tag{2.112}$$

This shaping constellation expansion ratio becomes infinite with N. For reference, the shaping gain $\gamma_s(\mathbb{R})$, shaping constellation expansion ratio $\mathrm{CER}_s(\mathbb{R})$, and peak-to-average power ratio $\mathrm{PAR}(\mathbb{R})$ of the N-sphere are shown in Table 2.1 for a number of values of N.

Table 2.1. γ_s, CER_s, and PAR for an N-Sphere

N	γ_s	$10\log_{10}\gamma_s$	CER_s	PAR
2	1.047	0.200	1.000	2
4	1.111	0.456	1.414	3
6	1.153	0.617	1.817	4
8	1.183	0.729	2.213	5
12	1.224	0.879	2.994	7
16	1.252	0.976	3.764	9
20	1.272	1.044	4.529	11
24	1.287	1.096	5.289	13
28	1.299	1.136	6.046	15
32	1.309	1.169	6.800	17
40	1.324	1.219	8.304	21
48	1.335	1.256	9.803	25
56	1.344	1.284	11.298	29
64	1.351	1.306	12.790	33
80	1.361	1.340	15.769	41

As an example, suppose a spherically shaped N-dimensional constellation is to be designed using the lattice $\mathbf{Z}^N$. Then according to (2.106), the number of points in the constituent 2D constellation is approximately

$$|\langle\mathbf{C}\rangle_2| \simeq \mathrm{CER}_s(\mathbb{R})\,|\mathbf{C}|^{2/N} = [(N/2)!]^{2/N}|\mathbf{C}|^{2/N} \tag{2.113}$$

If, in a particular communication system, the desired goal is to send $\beta = 7.5$ bits per two dimensions or $b = 15$ bits per four dimensions using a constellation of 2^{15} points from $\mathbf{Z}^4$, then the constituent 2D constellation must contain approximately $2^{1/2}2^{15/2} = 256$ points with PAR $\simeq 3$ and a shaping gain of about 0.46 dB. If $\mathbf{Z}^8$ is used to send $\beta = 7.25$ bits per two dimensions or $b = 4 \times 7.25 = 29$ bits per eight dimensions using 2^{29} points, the constituent 2D constellation must contain about $(4!)^{1/4}2^{29/4} \simeq 337$ points with PAR $\simeq 5$ and a shaping gain of about 0.73 dB. As a comparison, the easily implemented generalized cross constellation [25] for $N = 4$ and $\beta = 7.5$ has 192 points in its constituent 2D constellation. The generalized cross constellation for $N = 8$ and $\beta = 7.25$ has 160 points in its constituent 2D constellation. Both of these cross constellations have a PAR of about 2 and a shaping gain around 0.3 dB. The reduced shaping gain of these generalized cross constellations results in the smaller PAR and number of points in the constituent 2D constellations. ■

2.7 Factors of the Peak-to-Average Power Ratio

The continuous approximation can be used to represent the peak-to-average power ratio as a product of factors that give insight into constellation design. Consider a constellation consisting of the points $\mathbf{C} = \mathbf{A} \cap \mathbb{R}$. We will assume the points in $\mathbf{C}$ are used with equal likelihood and that points are uniformly distributed over $\mathbb{R}$. Then $\mathrm{PAR}(\mathbf{C})$ is approximately

$$\mathrm{PAR}(\mathbf{C}) \simeq \mathrm{PAR}(\mathbb{R}) = \frac{\max\limits_{\mathbf{x}\in\langle\mathbb{R}\rangle_2} \|\mathbf{x}\|^2}{\mathbb{P}(\mathbb{R})} \tag{2.114}$$

The points in the constituent 2D region $\langle\mathbb{R}\rangle_2$ are generally not uniformly distributed. However, it will be convenient to define the PAR for $\langle\mathbb{R}\rangle_2$ when the points are uniformly distributed to be

$$\mathrm{PAR}_u(\langle\mathbb{R}\rangle_2) = \frac{\max\limits_{\mathbf{x}\in\langle\mathbb{R}\rangle_2} \|\mathbf{x}\|^2}{\mathbb{P}_u(\langle\mathbb{R}\rangle_2)} \tag{2.115}$$

where $\mathbb{P}_u(\langle\mathbb{R}\rangle_2)$ is the average power of $\langle\mathbb{R}\rangle_2$ assuming a uniform distribution. So

$$\max_{\mathbf{x}\in\langle\mathbb{R}\rangle_2} \|\mathbf{x}\|^2 = \mathrm{PAR}_u(\langle\mathbb{R}\rangle_2)\mathbb{P}_u(\langle\mathbb{R}\rangle_2) \tag{2.116}$$

The shaping gain for $\mathbb{R}$ was defined as

$$\gamma_s(\mathbb{R}) = \frac{V^{2/N}(\mathbb{R})}{6\,\mathbb{P}(\mathbb{R})} \tag{2.117}$$

Similarly, the shaping gain for $\langle\mathbb{R}\rangle_2$ assuming a uniform point distribution is

$$\gamma_s^{(u)}(\langle\mathbb{R}\rangle_2) = \frac{V(\langle\mathbb{R}\rangle_2)}{6\,\mathbb{P}_u(\langle\mathbb{R}\rangle_2)} \tag{2.118}$$

Taking the ratio of these two shaping gains and rearranging gives

$$\frac{1}{\mathbb{P}(\mathbb{R})} = \frac{\gamma_s(\mathbb{R})}{\gamma_s^{(u)}(\langle\mathbb{R}\rangle_2)} \frac{V(\langle\mathbb{R}\rangle_2)}{V^{2/N}(\mathbb{R})} \frac{1}{\mathbb{P}_u(\langle\mathbb{R}\rangle_2)} \tag{2.119}$$

Multiplying (2.116) by (2.119) gives the following desired formula for the PAR

$$\begin{aligned} \mathrm{PAR}(\mathbb{R}) &= \mathrm{PAR}_u(\langle\mathbb{R}\rangle_2) \frac{\gamma_s(\mathbb{R})}{\gamma_s^{(u)}(\langle\mathbb{R}\rangle_2)} \frac{V(\langle\mathbb{R}\rangle_2)}{V^{2/N}(\mathbb{R})} \\ &= \mathrm{PAR}_u(\langle\mathbb{R}\rangle_2) \frac{\gamma_s(\mathbb{R})}{\gamma_s^{(u)}(\langle\mathbb{R}\rangle_2)} \mathrm{CER}_s(\mathbb{R}) \qquad (2.120) \\ &= \mathrm{PAR}_u(\langle\mathbb{R}\rangle_2)\,\mathrm{B}(\mathbb{R})\,\mathrm{CER}_s(\mathbb{R}) \qquad (2.121) \end{aligned}$$

where $\mathrm{B}(\mathbb{R})$ is the biasing gain defined by (2.83).

A primary reason for using multidimensional constellations is to achieve shaping gain. With good shaping, points in the constituent 2D constellation closer to the origin are more probable than those further away. Therefore, the average power normalized to two dimensions, $\mathbb{P}(\mathbb{R})$, will be less than the average power, $\mathbb{P}_u(\langle\mathbb{R}\rangle_2)$, when the points are uniformly distributed over the constituent 2D region, so $\mathrm{PAR}(\mathbb{R})$ will be greater than $\mathrm{PAR}_u(\langle\mathbb{R}\rangle_2)$. A goal of constellation design is to achieve good shaping gain while keeping the PAR as small as possible. These two quantities are related and depend on the geometry of the regions. For example, some values of the shaping gain and required PAR for an N-sphere are shown in Table 2.1. From (2.121) we observe that to keep the PAR low, The N-dimensional region $\mathbb{R}$ should have a low shaping constellation expansion ratio and the constituent 2D region should have a low PAR when its points are uniformly distributed.

EXAMPLE 2.10 Factorization of $\mathrm{PAR}(\mathbb{R})$ for an N-Sphere

According to (2.64) $\mathrm{PAR}_u(\langle\mathbb{R}\rangle_2) = 2$. From (2.94)

$$\gamma_s(\mathbb{R}) = \frac{\pi\left(\frac{N}{2}+1\right)}{6\left[\left(\frac{N}{2}\right)!\right]^{2/N}} \text{ for } N \text{ even so } \gamma_s^{(u)}(\langle\mathbb{R}\rangle_2) = \pi/3 \tag{2.122}$$

According to (2.112) $\mathrm{CER}_s(\mathbb{R}) = [(N/2)!]^{2/N}$. Substituting these quantities into (2.121) gives $\mathrm{PAR}(\mathbb{R}) = (N/2)+1$ which agrees with the value determined by (2.64). ∎

2.8 Optimum Tradeoffs of Shaping Gain with CER_s and PAR

We have seen that shaping gain can be achieved by using a multidimensional constellation consisting of 2^b equally likely points selected from an N-dimensional lattice and contained in an N-dimensional region more spherical than an N-cube. This typically induces a distribution on the constituent 2D constellations with points of low energy closer to the origin occurring with larger probability than high energy points farther from the origin. In particular, it was demonstrated in Section 2.5.3.1 that the constituent 2D pdf's induced by an N-sphere converge to a 2D circular Gaussian pdf highly concentrated around the origin for large N. The peak-to-average power ratio and shaping constellation expansion ratio become infinite with N for the N-sphere. However, the observation that points with large energy occur with low probability suggests that some N-dimensional points that induce 2D points with large energy could be eliminated from the N-dimensional constellation thereby reducing PAR and CER_s with negligible reduction of the shaping gain. Some results on optimum tradeoffs of shaping gain with PAR and CER_s derived by Forney and Wei [25] are summarized in this section.

Suppose all the constituent 2D constellations $\langle\mathbf{C}\rangle_2$ of the N-dimensional constellation $\mathbf{C}$ have the same induced probability distribution and that $\mathbf{C}$ has 2^b equally likely points. Let the probability mass distribution for points in $\langle\mathbf{C}\rangle_2$ be $p(\mathbf{x})$ where $\mathbf{x} = (x_1, x_2)$. Then the entropy for the constituent 2D constellation is defined as

$$H(\langle\mathbf{C}\rangle_2) = -\sum_{\mathbf{x}\in\langle\mathbf{C}\rangle_2} p(\mathbf{x}) \log_2 \mathrm{p}(\mathbf{x}) \tag{2.123}$$

bits per two dimensions. The entropy of $\mathbf{C}$ is b bits per N dimensions. It can be shown (see page 25 of Gallager [27]) that

$$\frac{N}{2} H(\langle\mathbf{C}\rangle_2) \geq H(\mathbf{C}) = b \tag{2.124}$$

with equality when $\mathbf{C}$ is the Cartesian product of $N/2$ versions of $\langle\mathbf{C}\rangle_2$ with independent identical distributions. Therefore, the entropy of the constituent 2D constellation is lower bounded by the normalized bit rate, that is,

$$H(\langle\mathbf{C}\rangle_2) \geq \frac{2}{N} b = \beta \tag{2.125}$$

The average power for the constituent 2D constellation is

$$P(\langle \mathbf{C} \rangle_2) = \sum_{\mathbf{x} \in \langle \mathbf{C} \rangle_2} (x_1^2 + x_2^2) p(\mathbf{x}) \tag{2.126}$$

With a little thought, it can be seen that the average power for the N-dimensional constellation $\mathbf{C}$ is $N/2$ times that of the constituent 2D constellation, so this power normalized to two dimensions is $\mathbb{P}(\mathbf{C}) = (2/N)P(\mathbf{C}) = P(\langle \mathbf{C} \rangle_2)$.

We have seen that the continuous approximation can be used for constellations with many points. In this case, the 2D probability mass distribution can be approximated by a continuous 2D probability density function $f(\mathbf{x})$. The differential entropy for a 2D random variable $\mathbf{X}$ with this density is

$$H(\mathbf{X}) = -\int f(\mathbf{x}) \log_2 f(\mathbf{x})\, d\mathbf{x} \tag{2.127}$$

The density that maximizes $H(\mathbf{X})$ for a fixed average signal power $\mathbb{P}(\mathbf{X}) = \mathrm{E}\{\|\mathbf{x}\|^2\}$ in two dimensions, or that minimizes $\mathbb{P}(\mathbf{X})$ for a fixed $H(\mathbf{X})$, is the two dimensional circular Gaussian pdf [5]

$$f(\mathbf{x}) = \frac{1}{2\pi\sigma^2} e^{-\frac{\|\mathbf{x}\|^2}{2\sigma^2}} \tag{2.128}$$

where $2\sigma^2 = \mathbb{P}(\mathbf{X})$. The corresponding differential entropy is $H_{\max}(\mathbf{X}) = \log_2(2\pi e\sigma^2)$ bits per two dimensions. A fundamental result in information theory [27] is that a constellation with entropy $H(\mathbf{X})$ can be used to transmit arbitrarily close to $H(\mathbf{X})$ bits per two dimensions by using a sufficiently powerful source code. The normalized baseline average power for the optimum Gaussian density and maximum achievable bit rate $\beta_{\max} = H_{\max}(\mathbf{X})$ is

$$\mathbb{P}_{\oplus}(\mathbf{X}) = \frac{2^{H_{\max}(\mathbf{X})}}{6} = \frac{2\pi e\sigma^2}{6} \tag{2.129}$$

The the average power required to send at this rate is $\mathbb{P}(\mathbf{X}) = 2\sigma^2$, so the shaping gain is

$$\gamma_s = \frac{\mathbb{P}_{\oplus}(\mathbf{X})}{\mathbb{P}(\mathbf{X})} = \frac{\pi e}{6} = 1.423 \ \textit{or} \ 1.53 \ \text{dB} \tag{2.130}$$

This same result was obtained in Section 2.5.3 for the N-sphere as N becomes large where the 1.53 dB shaping gain limit was obtained and the constituent 2D pdf was shown to converge to the circular 2D Gaussian pdf.

The PAR for a constellation with a Gaussian distribution is infinite. To obtain some relationships for constellations with finite PAR, let us restrict the constituent 2D constellation to be confined to a circular disc of radius R. Using

standard techniques from the calculus of variations, it can be shown that the pdf which is zero outside of a circle of radius R and maximizes the differential entropy $H(\mathbf{X})$ for a fixed average power $\mathbb{P}(\mathbf{X})$, or minimizes $\mathbb{P}(\mathbf{X})$ for a fixed $H(\mathbf{X})$, is the truncated circular Gaussian pdf

$$f(\mathbf{x}) = \begin{cases} \frac{\lambda}{\pi R^2(1-e^{-\lambda})} e^{-\lambda\|\mathbf{x}\|^2/R^2} & \text{for } \|\mathbf{x}\| \le R \\ 0 & \text{elsewhere} \end{cases} \tag{2.131}$$

where λ is a positive number that controls the tradeoff between the average power $\mathbb{P}_\lambda(\mathbf{X})$ and entropy $H_\lambda(\mathbf{X})$. As λ approaches zero, $f(\mathbf{x})$ becomes uniform over the circle of radius R. As λ becomes large, $f(\mathbf{x})$ converges to a zero mean Gaussian pdf of variance $R^2/(2\lambda)$. The average power and differential entropy in terms of λ are

$$\mathbb{P}_\lambda(\mathbf{X}) = R^2 \frac{g_2(\lambda)}{\lambda g_1(\lambda)} \tag{2.132}$$

and

$$H_\lambda(\mathbf{X}) = -\log_2\left[\frac{\lambda}{\pi R^2 g_1(\lambda)}\right] + \frac{g_2(\lambda)}{g_1(\lambda)} \log_2 e \tag{2.133}$$

where

$$g_1(\lambda) = 1 - e^{-\lambda} \tag{2.134}$$

$$g_2(\lambda) = g_1(\lambda) - \lambda e^{-\lambda} \tag{2.135}$$

The peak-to-average power ratio in terms of λ is

$$\mathrm{PAR}(\lambda) = \frac{R^2}{\mathbb{P}_\lambda(\mathbf{X})} = \lambda \frac{g_1(\lambda)}{g_2(\lambda)} \tag{2.136}$$

and the shaping gain in terms of λ is

$$\gamma_s(\lambda) = \frac{\mathbb{P}_{\oplus}(\mathbf{X})}{\mathbb{P}_\lambda(\mathbf{X})} = \frac{2^{H_\lambda(\mathbf{X})}/6}{\mathbb{P}_\lambda(\mathbf{X})} = \frac{\pi g_1^2(\lambda)}{6 g_2(\lambda)} e^{g_2(\lambda)/g_1(\lambda)} \tag{2.137}$$

Assuming the constellation points are chosen from an N-dimensional lattice with a fundamental volume of 1, the size of the constituent 2D lattice is the area of the circle, πR^2. The size of the baseline constituent 2D constellation is $2^{H_\lambda(\mathbf{X})}$, so the constellation expansion ratio is

$$\mathrm{CER}_s(\lambda) = \frac{\pi R^2}{2^{H_\lambda(\mathbf{X})}} = \frac{\lambda}{g_1(\lambda)} e^{-g_2(\lambda)/g_1(\lambda)} \tag{2.138}$$

We have seen that the PAR for a circle with a uniform distribution is 2 and its shaping gain is $\pi/3$. A direct computation shows that

$$\mathrm{PAR}(\lambda) = 2 \times \frac{\gamma_s(\lambda)}{\pi/3} \times \mathrm{CER}_s(\lambda) \tag{2.139}$$

This is another example of the factorization of the PAR given by (2.121). Thus, shaping increases the PAR of 2 for the unshaped constellation by the second factor which is the biasing gain and by the third factor $\mathrm{CER}_s(\lambda)$ resulting from the constellation expansion.

The tradeoff between shaping gain and PAR obtained from (2.137) and (2.136) is plotted in Figure 2.4. The tradeoff between shaping gain and $\mathrm{CER}_s(\lambda)$ given by (2.137) and (2.138) is plotted in Figure 2.5. From these figures, we see that it is possible to achieve a shaping gain of about 1.2 dB with $\mathrm{CER}_s \simeq 1.2$ and $\mathrm{PAR} \simeq 3$, for example. The PAR of 3 is the product of 2 for the PAR of the unshaped circular constellation, a biasing gain of 1.25, and the constellation expansion of 1.2.

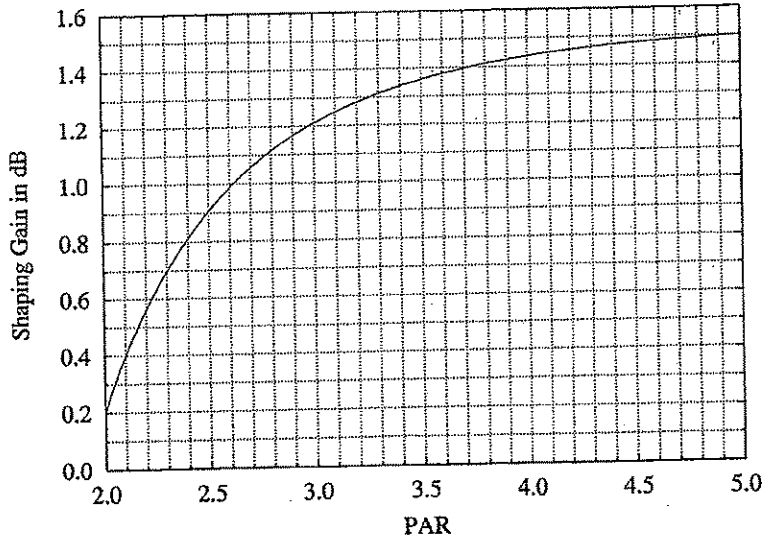


Figure 2.4. Best Tradeoff Between Shaping Gain and Peak-to-Average Ratio

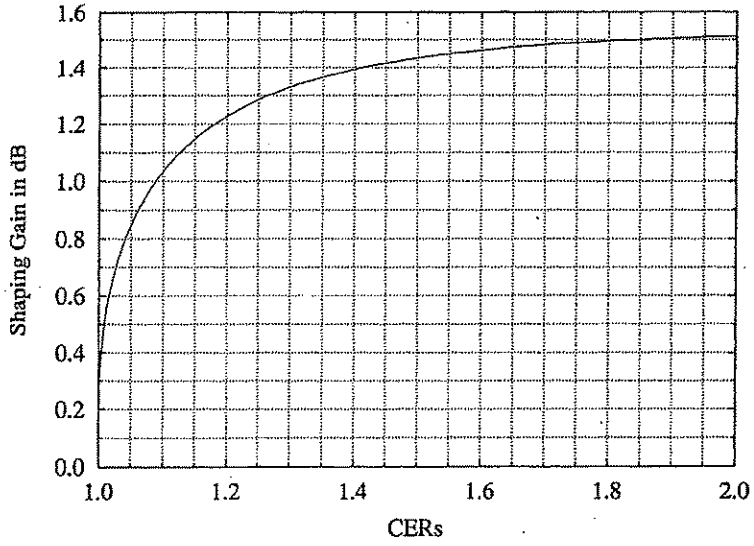


Figure 2.5. Best Tradeoff Between Shaping Gain and Constellation Expansion Ratio

Chapter 3

PRINCIPLES OF CONVOLUTIONAL AND TRELLIS CODES

The purpose of this chapter is to introduce basic definitions, notation, and concepts for binary convolutional codes and trellis codes as a reference for later chapters. For detailed treatments of these codes see Viterbi and Omura [65], Schlegel [57], and Johannesson and Zigangirov [37]. First, a tool for analyzing linear sequential circuits called the Huffman D-transform is presented. It is essentially the same as the Z-transform used in the digital signal processing field. Realizations for linear time-invariant sequential circuits that can be used as encoders are derived using D-transform methods. Then convolutional codes are described in terms of their generator matrices and polynomials, check matrices, and inverse check matrices. Conversion from non-systematic to systematic form is discussed. Convenient representations of code sequences in terms of a trellis diagram or a state transition diagram are introduced and the error correction properties of binary convolutional codes in terms of the weight distributions of the code sequences are briefly discussed. Next, the method of combining binary convolutional codes with QAM modulation, known as trellis coded modulation (TCM), is presented. Finally, a brief introduction to the Viterbi decoding algorithm is presented and a formula for the fundamental coding gain is given.

3.1 The Huffman D-Transform

The two-sided D-transform or Huffman transform of a sequence $f(n)$ is defined to be the power series

$$F(D) = \sum_{n=-\infty}^{\infty} f(n)D^n \tag{3.1}$$

When dealing with binary convolutional codes, the values of $f(n)$ can be one of two values, 0 or 1. In the digital signal processing field, it is customary to replace D by z^{-1} and call the series the Z-transform. These transforms allow signals to be compactly represented by rational functions of D and linear time-invariant systems to be analyzed by simple algebraic means in the transform domain.

EXAMPLE 3.1 Unit Step Function

The unit step function is defined to be

$$u(n) = \begin{cases} 1 & \text{for } n \geq 0 \\ 0 & \text{for } n < 0 \end{cases} \tag{3.2}$$

Then

$$U(D) = \sum_{n=0}^{\infty} D^n = \frac{1}{1-D} \quad \text{for } |D| < 1 \tag{3.3}$$

This is the sum of a geometric series with ratio D. The time sequence can be obtained from its D transform by expanding the transform into a power series by a variety of methods. In this case, the closed form for the sum of a geometric series is evident. The denominator, $1 - D$, can also be divided into the numerator, 1, to obtain the power series coefficients. ■

The one-sided D-transform is sometimes used to automatically include initial conditions stored in the sequential circuit. Let $f(n)$ be a sequence that may or may not be zero for $n < 0$. Its one-sided transform is defined to be

$$F_+(D) = \sum_{n=0}^{\infty} f(n) D^n \tag{3.4}$$

That is, the sum always begins at time $n = 0$.

3.1.1 Two-Sided Transform of a Delayed Sequence

Suppose $f(n)$ has the transform $F(D)$. Let $f(n)$ delayed by L samples be $g(n) = f(n - L)$. The delay L can be any positive or negative integer. Then $G(D) = D^L F(D)$.

Proof:

$$G(D) = \sum_{n=-\infty}^{\infty} g(n) D^n = \sum_{n=-\infty}^{\infty} f(n-L) D^n \tag{3.5}$$

Replacing $n - L$ by m gives

$$G(D) = \sum_{m=-\infty}^{\infty} f(m) D^{m+L} = D^L F(D) \tag{3.6}$$

3.1.2 One-Sided Transform of a Delayed Sequence

Let $f(n)$ have the one-sided transform $F_+(D)$ and let L be an integer greater than or equal to 0. The delayed sequence $g(n) = f(n - L)$ has the one-sided transform

$$G_+(D) = D^L F_+(D) + \sum_{n=0}^{L-1} f(n-L) D^n \tag{3.7}$$

The sum depends on the initial sequence values $f(-L), f(1-L), \ldots, f(-1)$.

Proof:

$$\begin{aligned} G_+(D) &= \sum_{n=0}^{\infty} g(n) D^n = \sum_{n=0}^{\infty} f(n-L) D^n \\ &= \sum_{n=0}^{L-1} f(n-L) D^n + \sum_{n=L}^{\infty} f(n-L) D^n \end{aligned} \tag{3.8}$$

Making the substitution $m = n - L$ in the last summation on the right gives

$$\begin{aligned} G_+(D) &= \sum_{n=0}^{L-1} f(n-L) D^n + \sum_{m=0}^{\infty} f(m) D^{m+L} \\ &= \sum_{n=0}^{L-1} f(n-L) D^n + D^L F_+(D) \end{aligned} \tag{3.9}$$

EXAMPLE 3.2

Consider a system whose input $x(n)$ and output $y(n)$ are related by the first-order difference equation $y(n) = x(n) + y(n-1)$. Taking the one-sided transform of both sides gives

$$Y_+(n) = X_+(n) + [y(-1) + DY_+(D)] \tag{3.10}$$

So

$$Y_+(D) = \frac{X_+(D)}{1-D} + \frac{y(-1)}{1-D} \tag{3.11}$$

The first term on the right is the output when the initial condition is $y(-1) = 0$. Let the zero initial condition solution be denoted by $y_0(n)$. The second term is the observed output caused by the initial condition when the input is identically 0. Therefore,

$$y(n) = y_0(n) + y(-1) \quad \text{for } n \geq 0 \tag{3.12}$$

■

3.1.3 D-Transform of a Convolution

The convolution of two sequences $f(n)$ and $g(n)$ is defined to be the sequence

$$h(n) = \sum_{k=-\infty}^{\infty} f(k)g(n-k) = \sum_{k=-\infty}^{\infty} g(k)f(n-k) \tag{3.13}$$

The D-transform of the convolution is

$$H(D) = F(D)G(D) \tag{3.14}$$

Proof:

$$\begin{aligned} H(D) &= \sum_{n=-\infty}^{\infty} \left[\sum_{k=-\infty}^{\infty} f(k)g(n-k)D^n \right] \\ &= \sum_{k=-\infty}^{\infty} f(k) \sum_{n=-\infty}^{\infty} g(n-k)D^n \end{aligned} \tag{3.15}$$

The last summation on the right is just the D-transform of the delayed sequence $g(n-k)$, so using the delay property (3.5) gives

$$\begin{aligned} H(D) &= \sum_{k=-\infty}^{\infty} f(k)G(D)D^k = G(D) \sum_{k=-\infty}^{\infty} f(k)D^k \\ &= F(D)G(D) \end{aligned} \tag{3.16}$$

3.2 Transfer Functions and Realizations

The output $y(n)$ of a linear, time-invariant, sequential circuit is the convolution of its input $x(n)$ with its unit pulse response $h(n)$, that is

$$y(n) = \sum_{k=-\infty}^{\infty} h(k)x(n-k) \tag{3.17}$$

For binary circuits, the signal values can only be 0 or 1 and modulo 2 addition is used in the convolution formula. The D-transform of the output is

$$Y(D) = X(D)H(D) \tag{3.18}$$

The *transfer function* for the circuit is defined to be the ratio

$$H(D) = Y(D)/X(D) \tag{3.19}$$

and is the transform of the circuit's unit pulse response. Initial conditions are all assumed to be 0 when computing transfer functions.

EXAMPLE 3.3 Transfer Function of a Delay Element

Suppose the input and output of a circuit are related by the equation $y(n) = x(n-1)$. Then $Y(D) = DX(D)$ and the transfer function is $H(D) = Y(D)/X(D) = D$. Therefore, the symbol D is often used to represent a one unit delay element. ∎

Linear, time-invariant sequential circuits with a finite number of storage elements have transfer functions that are the ratio of two polynomials in D. The ratio of two polynomials is said to be a *rational function*. We will assume that the rational transfer function has the form

$$H(D) = \frac{A(D)}{B(D)} = \frac{a_0 + a_1 D + \cdots + a_M D^M}{1 + b_1 D + \cdots + b_N D^N} \tag{3.20}$$

The numerator order M can be less than, greater than, or equal to the denominator order N. When the denominator is $B(D) = 1$, the circuit is called a *finite duration impulse response* (FIR) system. When $B(D)$ is not 1, the circuit is called a *recursive* or *infinite duration impulse response* (IIR) system.

A rational transfer function can be realized by many different circuits. Two common realizations will be presented in the following sections.

3.2.1 Type 1 Direct Form Realization

The rational transfer function can be decomposed into the cascade of the denominator and numerator portions as shown in Figure 3.1. The transform

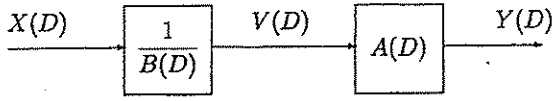


Figure 3.1. Representing $H(D)$ as a Cascade

of the intermediate signal $V(D)$ is related to the input $X(D)$ by the equation $V(D) = X(D)/B(D)$. Therefore,

$$\begin{aligned} X(D) &= B(D)V(D) = (1 + b_1 D + \cdots + b_N D^N)V(D) \\ &= V(D) + b_1 V(D)D + \cdots + b_N V(D)D^N \end{aligned} \tag{3.21}$$

or

$$V(D) = X(D) - b_1 V(D)D - \cdots - b_N V(D)D^N \tag{3.22}$$

In the time domain, this is equivalent to the following difference equation:

$$v(n) = x(n) - b_1 v(n-1) - \cdots - b_N v(n-N) \tag{3.23}$$

Similarly,

$$Y(D) = A(D)V(D) = a_0 V(D) + a_1 V(D)D + \cdots + a_M V(D)D^M \tag{3.24}$$

or

$$y(n) = a_0 v(n) + a_1 v(n-1) + \cdots + a_M v(n-M) \tag{3.25}$$

These two equations describe what we will call a *type 1 direct form* realization.The nomenclature is not standard and this realization is sometimes called the *controller canonical form*. A block diagram for this realization is shown in Figure 3.2 with $M = N$. This incurs no loss of generality because the appropriate higher order coefficients can be set to zero when M and N differ. It is called a direct form because the transfer function coefficients appear explicitly in the equations or block diagram.

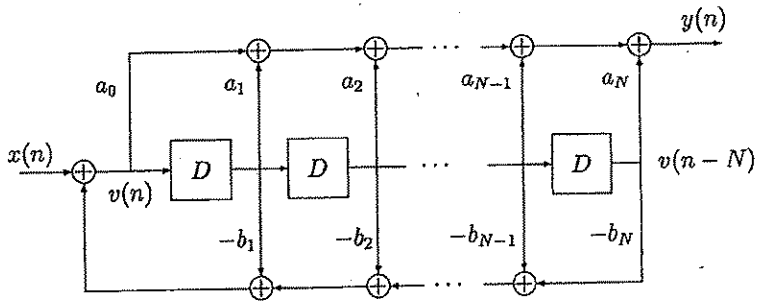


Figure 3.2. Type 1 Direct Form Realization

3.2.2 Type 2 Direct Form Realization

A second realization know as the *type 2 direct form* or *observer canonical form* will now be developed. We will let $M = N$ for simplicity. We know that

$$H(D) = \frac{A(D)}{B(D)} = \frac{Y(D)}{X(D)} \tag{3.26}$$

Cross multiplying gives

$$Y(D)B(D) = X(D)A(D) \tag{3.27}$$

or

$$Y(D)(1 + b_1 D + \cdots + b_N D^N) = X(D)(a_0 + a_1 D + \cdots + a_N D^N) \tag{3.28}$$

This can be rearranged into the form

$$Y(D) = a_0 X(D) + \sum_{k=1}^{N} [a_k X(D) - b_k Y(D)]D^k \tag{3.29}$$

The block diagram for the realization suggested by this equation is shown in Figure 3.3

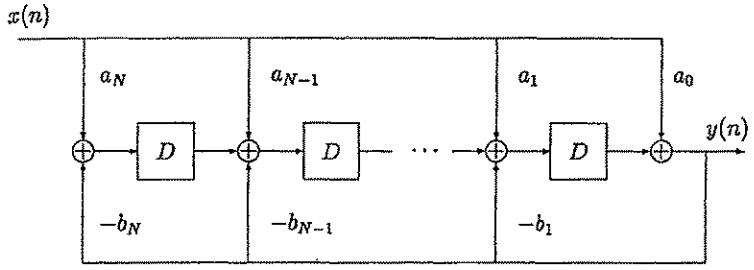


Figure 3.3. Type 2 Direct Form Realization

3.3 Description of a Convolutional Code by its Generator Matrix

An (N, K) convolutional encoder is a device that accepts successive input blocks of K bits from a data source, adds redundancy to the input blocks, and outputs blocks of N coded bits where $N > K$ in such a way that some errors introduced in the output stream when transmitted over a channel can be corrected. The current output block is a function of the present and some past input blocks. The output may depend on a finite or infinite number of past inputs. When the current output block only depends on the current input block, the code is called a *block code* and is a special case of a convolutional code with no memory. The *code rate* is defined as the ratio $R = K/N$. Generally, the lower the code rate, the higher the error correction capability.

To make the notation concrete, we will envision that the serial data input bit stream is separated into the K signals $x_1(n), \ldots, x_K(n)$. These can be combined into the row vector

$$\mathbf{x}(n) = [x_1(n)\ x_2(n)\ \cdots\ x_K(n)] \tag{3.30}$$

Similarly, the encoder output blocks will be represented by the row vector

$$\mathbf{y}(n) = [y_1(n)\ y_2(n)\ \cdots\ y_N(n)] \tag{3.31}$$

Linear, time-invariant codes form an important class of convolutional codes. For these codes, the output code streams are sums of convolutions of the input

data streams with the impulse responses of a set of filters known as the *generators* for the code. The impulse response samples can only have the values 0 and 1 and sums in the convolutions are computed using modulo 2 addition. That is, the sum of two bits is their exclusive-or. In the coding literature, this is also called GF(2) arithmetic where GF stands for Galois field [67]. (It should be pointed out that there are some nonlinear time-invariant convolutional codes in common use such as the V.32 8-state code and the V.34 64-state code.) If $g_{i,j}(D)$ is the transfer function from the ith input stream to the jth output stream, the Huffman transform of the encoder output can be expressed as

$$\begin{aligned}\mathbf{Y}(D) &= \mathbf{X}(D)\begin{bmatrix} g_{1,1}(D) & g_{1,2}(D) & \cdots & g_{1,N}(D) \\ g_{2,1}(D) & g_{2,2}(D) & \cdots & g_{2,N}(D) \\ \vdots & & & \vdots \\ g_{K,1}(D) & g_{K,2}(D) & \cdots & g_{K,N}(D) \end{bmatrix} \\ &= \mathbf{X}(D)\mathbf{G}(D) \end{aligned} \tag{3.32}$$

When transforms are added, the coefficients of like powers of D are combined using GF(2) arithmetic. In general, these systems are multi-input and multi-output systems. The question of how to realize these systems most efficiently has been studied extensively by system theorists. For example, the question of how to realize a system with the minimum number of state variables has been answered [37].

The matrix $\mathbf{G}(D)$ is called the *generator matrix* for the code. When the components of $\mathbf{G}(D)$ are polynomials with maximum degree M, the convolutional encoder has finite memory and the output depends only on the present and past M inputs. The components can also be rational functions of D and then the encoder has infinite memory. The rank of $\mathbf{G}(D)$ must be K so that unique input sequences result in unique output sequences.

The contents of each delay element in an encoder realization can be selected as a *state variable* and a vector consisting of the set of state variables is called the *state* of the encoder. When a realization for an encoder has L delay elements, the state vector can take on 2^L values. The output for a time-invariant sequential circuit is a function of the current state and inputs. The next state is also a function of the current state and inputs. These functions may be linear or nonlinear but do not depend on the time index n.

EXAMPLE 3.4 The Ungerboeck 4-State Code

The Ungerboeck 4-state convolutional encoder shown in Figure 3.4 has $K = 1$ input stream and $N = 2$ output streams. The input-output relationship is

$$[y_1(D)\ y_2(D)] = x_1(D)[D\ \ 1 + D^2] \tag{3.33}$$

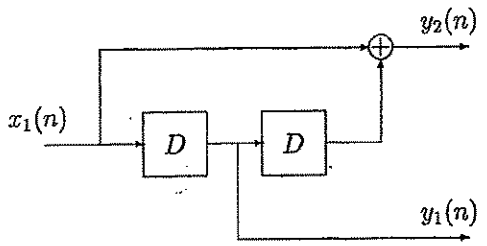


Figure 3.4. The Ungerboeck 4-State Convolutional Encoder

Thus the generator polynomials are

$$g_{1,1}(D) = D \text{ and } g_{1,2}(D) = 1 + D^2 \tag{3.34}$$

The time domain formulas for the outputs are

$$y_1(n) = x_1(n-1) \text{ and } y_2(n) = x_1(n) + x_1(n-2) \tag{3.35}$$

3.4 Systematic Form of a Convolutional Code

A systematic code is one in which the K input data streams appear explicitly in K of the N encoder output streams.The remaining $N-K$ output streams are called *check symbols*. A convolutional code with a $\mathbf{G}(D)$ matrix of rank K but arbitrary otherwise is not necessarily systematic. However, a non-systematic code can always be converted into an equivalent systematic one. Since $\mathbf{G}(D)$ has rank K, it must contain K linearly independent columns. These K columns can be moved to the right-hand side of $\mathbf{G}(D)$ by reordering the encoder output streams. Assuming this has been done, the generator matrix can be partitioned as follows:

$$\begin{aligned}\mathbf{G}(D) &= [\mathbf{A}(D)_{K\times(N-K)} \vdots \mathbf{B}(D)_{K\times K}] \\ &= \mathbf{B}(D)\mathbf{B}^{-1}(D)\,[\mathbf{A}(D) \vdots \mathbf{B}(D)] \\ &= \mathbf{B}(D)\,[\mathbf{B}^{-1}(D)\mathbf{A}(D) \vdots \mathbf{I}_{K\times K}] \end{aligned} \tag{3.36}$$

The encoder output is

$$\begin{aligned}\mathbf{Y}(D) &= \mathbf{X}(D)\mathbf{G}(D) = \mathbf{X}(D)\mathbf{B}(D)\,[\mathbf{B}^{-1}(D)\mathbf{A}(D) \vdots \mathbf{I}] \\ &= \hat{\mathbf{X}}(D)\,[\mathbf{B}^{-1}(D)\mathbf{A}(D) \vdots \mathbf{I}] = \hat{\mathbf{X}}(D)\hat{\mathbf{G}}(D) \end{aligned} \tag{3.37}$$

where

$$\hat{\mathbf{G}}(D) = [\mathbf{B}^{-1}(D)\mathbf{A}(D) \vdots \mathbf{I}] = [\mathbf{P}(D) \vdots \mathbf{I}] \tag{3.38}$$

and

$$\hat{\mathbf{X}}(D) = \mathbf{X}(D)\mathbf{B}(D) \tag{3.39}$$

Since $\mathbf{B}(D)$ is invertible, there is a unique $\hat{\mathbf{X}}(D)$ for each $\mathbf{X}(D)$ and vice versa. The set of codewords generated by an encoder with the generator matrix $\hat{\mathbf{G}}(D)$ is the same as by one with the generator matrix $\mathbf{G}(D)$. However, input data sequences are mapped to codewords differently. Since the set of codewords is the same for both codes, the probability of making an error between two sequences is the same in both cases. However, the bit error probability in the decoded outputs may be different.

The output of an encoder with input $\hat{\mathbf{X}}(D)$ and generator matrix $\hat{\mathbf{G}}(D)$ is

$$\mathbf{Y}(D) = [\hat{\mathbf{X}}(D)\mathbf{P}(D) \vdots \hat{\mathbf{X}}(D)] \tag{3.40}$$

Notice that the input sequence appears on the right-hand side of the encoded sequence, so the code is systematic. A block diagram for and encoder of this form is shown in Figure 3.5. There is nothing special about putting the input streams on the right. The initial generator matrix columns could have been rearranged to put them in any K columns.

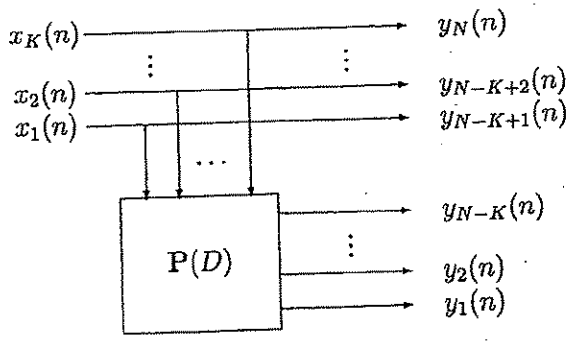


Figure 3.5. Block Diagram for a Systematic Convolutional Encoder

EXAMPLE 3.5 Systematic Form for Ungerboeck 4-State Code

The (2,1) 4-state Ungerboeck non-systematic code has the generator matrix

$$\mathbf{G}(D) = [D \quad 1 + D^2] \tag{3.41}$$

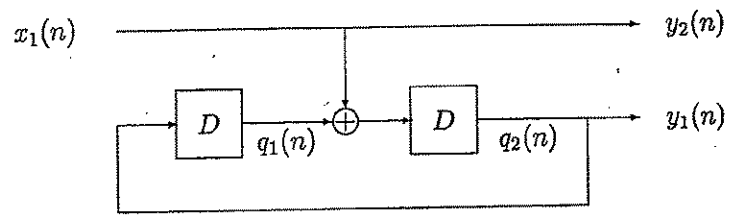


Figure 3.6. Block Diagram for Systematic Form of Ungerboeck 4-State Encoder

Let $\mathbf{A}(D) = D$ and $\mathbf{B}(D) = 1+D^2$, so $\mathbf{B}^{-1}(D) = 1/(1+D^2)$. The generator matrix for the equivalent systematic code is

$$\hat{\mathbf{G}}(D) = \begin{bmatrix} \dfrac{D}{1+D^2} & 1 \end{bmatrix} \tag{3.42}$$

A block diagram for this encoder using the type 2 direct form is shown in Figure 3.6. ■

3.5 The Parity Check Matrix and Syndromes

Consider an (N,K) systematic convolutional code with generator matrix $\mathbf{G}(D) = [\mathbf{P}(D) \vdots \mathbf{I}_{K\times K}]$. The *check symbols* in the output code block are

$$[Y_1(D), \ldots, Y_{N-K}(D)] = \mathbf{X}(D)\mathbf{P}(D) \tag{3.43}$$

and the *information symbols* are

$$[Y_{N-K+1}(D), \ldots, Y_N(D)] = \mathbf{X}(D) \tag{3.44}$$

Thus the check symbols can also be expressed as

$$[Y_1(D), \ldots, Y_{N-K}(D)] = [Y_{N-K+1}(D), \ldots, Y_N(D)]\,\mathbf{P}(D) \tag{3.45}$$

so

$$\begin{aligned}[Y_1(D), \ldots, Y_{N-K}(D)] - [Y_{N-K+1}(D), \ldots, Y_N(D)]\,\mathbf{P}(D) \\ = [0, \ldots, 0]_{1\times(N-K)}\end{aligned} \tag{3.46}$$

or, equivalently,

$$[Y_1(D) \cdots Y_N(D)] \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \\ 0 & 0 & \cdots & 1 \\ \hline & -\mathbf{P}(D) & & \end{bmatrix} = [0\,0 \cdots 0] \tag{3.47}$$

Note that with GF(2) arithmetic the minus sign in front of $\mathbf{P}$ can be left out. More compactly,

$$\mathbf{Y}(D)\mathbf{H}^t(D) = \mathbf{0} \tag{3.48}$$

where the superscript t stands for transpose and the $N-K \times K$ *parity check matrix* is defined to be

$$\mathbf{H}(D) = [\mathbf{I}_{(N-K)\times(N-K)} \vdots -\mathbf{P}^t(D)] \tag{3.49}$$

The parity check equation has a very simple interpretation. The check symbols are recomputed from the information symbols found in the last K positions of the code block and subtracted from the check symbols contained in the first $N-K$ positions in the block. If the observed block contains no error, the difference must be 0. Every codeword satisfies the check equation and every vector that satisfies the check equation is a codeword.

Suppose a codeword $\mathbf{Y}(D)$ is transmitted and $\mathbf{R}(D) = \mathbf{Y}(D) + \mathbf{E}(D)$ is received where $\mathbf{E}(D)$ is the channel error sequence. The *syndrome* for the received sequence is defined to be

$$\mathbf{S}(D) = [S_1(D), \ldots, S_{N-K}(D)] = \mathbf{R}(D)\mathbf{H}^t(D) \tag{3.50}$$

Substituting for $\mathbf{R}(D)$ we find that

$$\begin{aligned} \mathbf{S}(D) &= [\mathbf{Y}(D) + \mathbf{E}(D)]\mathbf{H}^t(D) = \mathbf{Y}(D)\mathbf{H}^t(D) + \mathbf{E}(D)\mathbf{H}^t(D) \\ &= \mathbf{E}(D)\mathbf{H}^t(D) \end{aligned} \tag{3.51}$$

Observe that the syndrome depends only on the channel error pattern. It does not depend on the transmitted codeword at all.

Now consider a new error pattern $\mathbf{E}'(D) = \mathbf{E}(D) + \mathbf{Z}(D)$ where $\mathbf{Z}(D)$ is any codeword. The syndrome for this new error pattern is

$$\mathbf{S}'(D) = \mathbf{E}'(D)\mathbf{H}^t(D) = \mathbf{E}(D)\mathbf{H}^t(D) = \mathbf{S}(D) \tag{3.52}$$

Thus, there are many error patterns that have the same syndrome. This set is said to form an equivalence class of error patterns.

EXAMPLE 3.6 Check Matrix for Ungerboeck 4-State Code

From the systematic form of the generator matrix for the Ungerboeck 4-state code given by (3.42), it can be seen that $\mathbf{P}(D) = D/(1+D^2)$. Therefore, a check matrix for this code is

$$\mathbf{H}(D) = \begin{bmatrix} 1 & \dfrac{D}{1+D^2} \end{bmatrix} \tag{3.53}$$

To verify that this is a check matrix, let $\mathbf{Y}(D)$ be any codeword. Then

$$\begin{aligned} \mathbf{Y}(D)\mathbf{H}^t(D) &= \mathbf{X}(D)\begin{bmatrix} \dfrac{D}{1+D^2} & 1 \end{bmatrix}\begin{bmatrix} 1 \\ \dfrac{D}{1+D^2} \end{bmatrix} \\ &= 2\mathbf{X}(D)\frac{D}{1+D^2} = 0 \end{aligned} \tag{3.54}$$

using GF(2) arithmetic for coefficients. Also

$$(1+D^2)\mathbf{Y}(D)\mathbf{H}^t(D) = \mathbf{Y}(D)\begin{bmatrix} 1+D^2 \\ D \end{bmatrix} = 0 \tag{3.55}$$

Therefore, another check matrix for this code is

$$\hat{\mathbf{H}}^t(D) = [1+D^2 \quad D] \tag{3.56}$$

In general, $\mathbf{H}(D)$ can always be multiplied by a polynomial to clear the fractions resulting in a check matrix whose entries are polynomials. ∎

3.6 Inverse Check Matrix or Inverse Syndrome Former

In Section 3.5 we observed that there is an equivalence class of received words that all have the same syndrome. The problem to be addressed in this section is to find one member of the class, which will be called the class representative, given the syndrome. A system for doing this is called an inverse syndrome former and can be represented by an inverse check matrix. The class representative can be expressed as

$$\mathbf{E}(D) = \mathbf{S}(D)[\mathbf{H}^t(D)]^{-1} \tag{3.57}$$

Since $\mathbf{H}(D)$ is not a square matrix, its inverse is not unique. One solution will be presented here. The rank of $\mathbf{H}(D)$ is $N-K$ so it must contain a set of $N-K$ linearly independent columns and $\mathbf{H}^t(D)$ must contain a set of $N-K$ linearly independent rows. Suppose these rows are in positions $i_1, i_2, \ldots, i_{N-K}$ and the rows are $\mathbf{c}_1(D), \ldots, \mathbf{c}_{N-K}(D)$. In other words, $\mathbf{H}^t(D)$ has the form

$$\mathbf{H}^t = \begin{bmatrix} \vdots \\ \mathbf{c}_1(D) \\ \vdots \\ \mathbf{c}_2(D) \\ \vdots \\ \mathbf{c}_{N-K}(D) \\ \vdots \end{bmatrix} \tag{3.58}$$

Let the $N-K$ independent rows be arranged in the $(N-K)\times(N-K)$ matrix

$$\mathbf{C}(D)=\begin{bmatrix}\mathbf{c}_1(D)\\ \mathbf{c}_2(D)\\ \vdots\\ \mathbf{c}_{N-K}(D)\end{bmatrix} \tag{3.59}$$

and designate its inverse by

$$\mathbf{C}^{-1}(D)=[\mathbf{d}_1(D)\ \mathbf{d}_2(D)\ \cdots\ \mathbf{d}_{N-K}(D)] \tag{3.60}$$

where the $\mathbf{d}$'s are column vectors. We want to find a sequence $\mathbf{E}(D)$ such that $\mathbf{E}(D)\mathbf{H}^t(D)=\mathbf{S}(D)$. Assume $\mathbf{E}(D)$ has the form

$$\mathbf{E}(D)=[0\ e_{i_1}(D)\ \mathbf{0}\ e_{i_2}(D)\ \mathbf{0}\ \cdots\ \mathbf{0}\ e_{i_{N-K}}(D)\ 0] \tag{3.61}$$

Then, because of the 0's in $\mathbf{E}(D)$,

$$\begin{aligned}\mathbf{E}(D)\mathbf{H}^t(D) &= [e_{i_1}(D)\ e_{i_2}(D)\ \cdots\ e_{i_{N-K}}(D)]\begin{bmatrix}\mathbf{c}_1(D)\\ \mathbf{c}_2(D)\\ \vdots\\ \mathbf{c}_{N-K}(D)\end{bmatrix}\\ &= \mathbf{S}(D)\end{aligned} \tag{3.62}$$

so the $N-K$ nonzero components of $\mathbf{E}(D)$ must be

$$[e_{i_1}(D)\ e_{i_2}(D)\ \cdots\ e_{i_{N-K}}(D)]=\mathbf{S}(D)\mathbf{C}^{-1}(D) \tag{3.63}$$

The zeros in $\mathbf{E}(D)$ can be obtained by putting zero columns in $\mathbf{C}^{-1}(D)$ as follows

$$\mathbf{E}(D)=\mathbf{S}(D)[0\ \mathbf{d}_1(D)\ \mathbf{0}\ \mathbf{d}_2(D)\ \mathbf{0}\ \cdots\ \mathbf{0}\ \mathbf{d}_{N-K}(D)\ 0] \tag{3.64}$$

Thus, one form of the inverse check matrix is

$$[\mathbf{H}^t(D)]^{-1}=[0\ \mathbf{d}_1(D)\ \mathbf{0}\ \mathbf{d}_2(D)\ \mathbf{0}\ \cdots\ \mathbf{0}\ \mathbf{d}_{N-K}(D)\ 0] \tag{3.65}$$

With this choice, we see that

$$[\mathbf{H}^t(D)]^{-1}\mathbf{H}^t(D)=\mathbf{I}_{(N-K)\times(N-K)} \tag{3.66}$$

In conclusion, using the method described above, a class representative with K zero components can always be found.

EXAMPLE 3.7 Inverse Check Matrix for Ungerboeck 4-State Code

According to (3.56) a transposed check matrix for this code is

$$\mathbf{H}^t(D)=\begin{bmatrix}1+D^2\\ D\end{bmatrix} \tag{3.67}$$

so we can let $i_1=1$ and

$$\mathbf{C}(D)=[1+D^2]\quad\text{and}\quad\mathbf{C}^{-1}(D)=\begin{bmatrix}\frac{1}{1+D^2}\end{bmatrix} \tag{3.68}$$

The corresponding inverse check matrix is

$$[\mathbf{H}^t(D)]^{-1}=\begin{bmatrix}\frac{1}{1+D^2}\ 0\end{bmatrix} \tag{3.69}$$

The syndrome $\mathbf{S}(D)$ is a scalar and the components of the class representative are

$$e_1(D)=\mathbf{S}(D)\times\frac{1}{1+D^2}\quad\text{and}\quad e_2(D)=0 \tag{3.70}$$

■

3.7 The Code Trellis

The *state* of a sequential circuit is a set of circuit variables such that the current output and next state can be computed from the current input and current state. The contents of the delay elements in a realization of a circuit can be used as a set of state variables. For example, the variables $q_1(n)$ and $q_2(n)$ shown in Figure 3.6 can be used as the state variables for the Ungerboeck systematic 4-state code.

The behavior of the circuit can be described by a *trellis diagram*. In a trellis diagram, the states are displayed as a vertical array of nodes which is repeated horizontally for each time instant. The state transitions are shown as lines connecting the nodes in adjacent arrays with circuit outputs labeling the lines. As an example, a section of the trellis diagram for the Ungerboeck systematic 4-state code is shown in Figure 3.7. This encoder has a single input. The transitions caused by a 0 input are shown as solid lines and those caused by a 1 input as dotted lines. In general, the inputs direct the encoder along a path in the trellis.

The trellis diagram can be condensed into a *state transition diagram*. Each state is shown as a node and the possible transitions between states are shown as curved lines with arrows. The encoder outputs are shown next to the lines. An example for the Ungerboeck systematic 4-state code is shown in Figure 3.8.

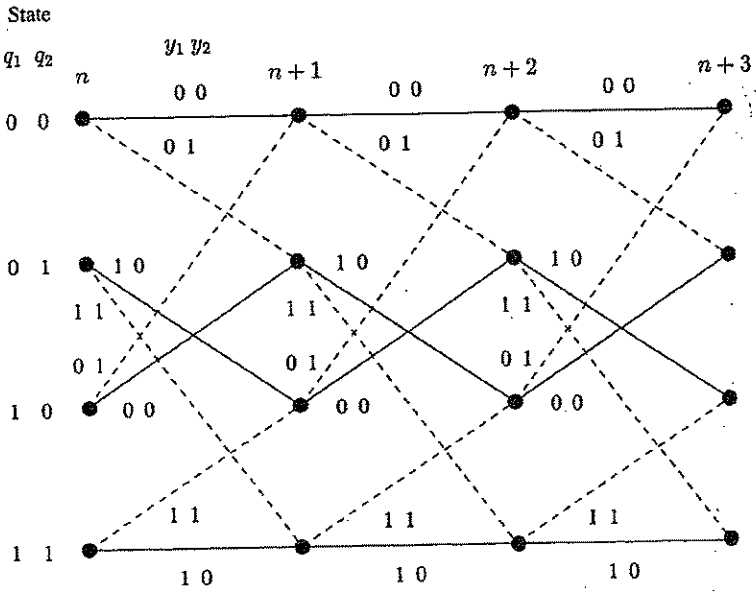


Figure 3.7. Trellis Diagram for the Ungerboeck Systematic 4-State Code

The numbers in the circles represent the states $[q_1(n), q_2(n)]$ and the numbers next to the lines are the outputs $[y_1(n), y_2(n)]$. Sometimes the input causing a transition is indicated by putting a slash after the outputs and then writing the input value. Since the Ungerboeck example is a systematic code, the output $y_2(n)$ is always equal to the input $x_1(n)$ and no extra information is required.

3.8 Weight Distributions and Error Correction Properties

The task of a decoder for a convolutional code can be viewed as estimating the path taken through the code trellis by the encoder based on the observed noise corrupted received signal. The Viterbi algorithm [64, 65] is the most popular decoding method for decoding convolutional codes. It makes a maximum likelihood estimate of the encoder's trellis path. Several types of errors are of interest when analyzing the performance of a decoder. One type called a *sequence error* occurs when the decoder follows a trellis path that deviates from the encoder's path. A special type of sequence error is a *first-event error* which is when the decoder excludes the correct path for the first time at depth j into the trellis. The ultimate goal of a decoder is to form an estimate of the information sequence entering the encoder. Errors in this estimate are called *bit errors*. The probabilities of these types of errors depend on the code structure,

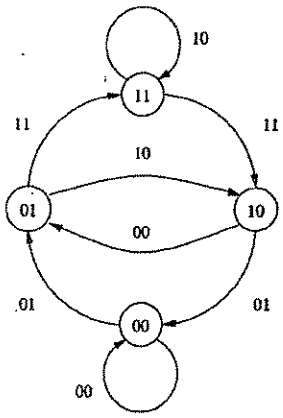


Figure 3.8. State Transition Diagram for the Ungerboeck Systematic 4-State Code

channel characteristics, and decoding algorithm. Exact formulas for these error probabilities are not known but moderately tight upper bounds have been found for some cases.

When the encoder output sequences are transmitted serially over a binary symmetric channel (BSC) or additive white Gaussian noise channel (AWGN) with binary phase-shift keying (BPSK), the error probabilities depend on the Hamming distances between the transmitted code bit sequence along a trellis path and sequences that diverge from the correct path at some point and later remerge with it. The *Hamming distance* between two binary sequences of the same length is the number of positions in which they differ. The *Hamming weight* of a sequence is the number of 1's in it. The number of sequences n_d at distance d from the correct one is also of interest. This set of numbers is called the *weight distribution*. For linear codes the all 0 sequence can be taken as the transmitted codeword without loss of generality since the weight distribution is the same relative to any codeword.

The weight distribution and additional information can be determined from a modified state transition diagram. An example for the Ungerboeck systematic 4-state code is shown in Figure 3.9. The 0 state is split in two with one part on the left and one on the right. Each branch is labeled with a D whose exponent is the number of 1's in the code sequence for that branch, an N with an exponent that is the number of 1's in the information sequence entering the encoder for the branch, and an L. Any path from the 0 state on the left to the 0 state on the right is a path that diverges from the all 0's path and remerges with it later for all time. The product of the branch labels along a path has an exponent of D equal to the number of 1's in the code sequence along the path, an exponent of L

equal to the path length in branches, and an exponent of N equal to the number of 1's in the input data sequence directing the encoder along that path. The sum of the products for all possible paths between the two 0 states is just the transfer function of the enhanced state transition diagram when it is considered to be a flow graph. Viterbi calls this function $T(D, L, N)$.

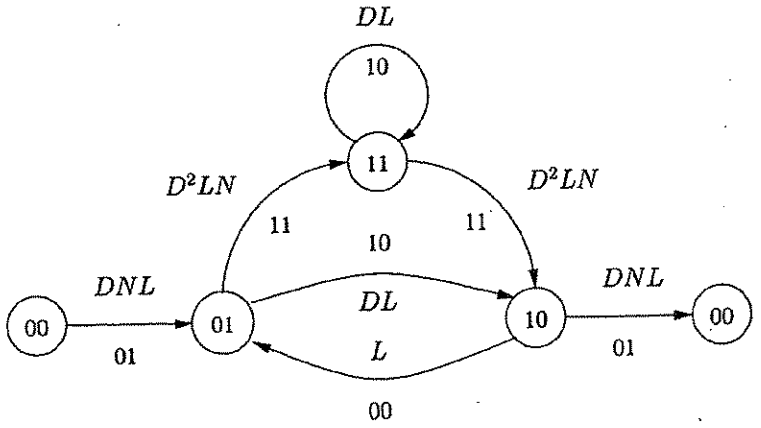


Figure 3.9. Opened and Enhanced State Transition Diagram for the Ungerboeck Systematic 4-State Code

EXAMPLE 3.8 Weight Distribution Generating Function for the Ungerboeck Systematic 4-State Encoder

Using the standard techniques for reducing a flow graph, the transfer function from left to right for Figure 3.9 is found to be

$$T(D,L,N) = D^3L^3N^2\frac{1-DL+D^3LN^2}{1-DL(1+L)+D^2L^3-D^4L^3N^2} \tag{3.71}$$

Dividing the denominator into the numerator, we find that the first few terms in the series expansion of the transfer function are

$$\begin{aligned} T(D,L,N) &= D^3L^3N^2 + D^4L^5N^2 + D^5L^7N^2 \\ &\quad + D^6(L^4N^4 + L^9N^2) + \cdots \end{aligned} \tag{3.72}$$

This sequence shows that there is one code sequence with Hamming weight 3, with a length of 3 branches, and caused by two 1 inputs. There is one code sequence with Hamming weight 4, length 5, and caused by two 1 inputs. The next term corresponds to a weight 5 path of length 7 caused by two 1 inputs.

The last term shows there are two sequences of weight 6 – one of length 4 cause by four 1 inputs and one of length 9 caused by two 1 inputs. You should look for these paths in the enhanced state transition diagram. ■

Viterbi [64, 65] has shown that when the code bits are transmitted serially over a binary symmetric channel with cross-over probability p, the first-event error probability for a Viterbi decoder can be upper bounded in terms of the weight distribution generating function by

$$P_E < T(D,L,N)\big|_{L=N=1,D=2\sqrt{p(1-p)}} \tag{3.73}$$

The bit error probability in the decoded information sequence is upper bounded by

$$P_B < \frac{\partial T(D,L,N)}{\partial N}\bigg|_{L=N=1,D=2\sqrt{p(1-p)}} \tag{3.74}$$

When the code bits are transmitted over an additive white Gaussian noise channel with one-sided noise power spectral density N_0 using binary phase shift keying with energy ϵ_s per transmitted code bit, the first-event error probability is upper bounded by

$$P_E < \operatorname{erfc}\sqrt{\frac{2d\epsilon_s}{N_0}}\exp\left(\frac{d\epsilon_s}{N_0}\right)T(D,L,N)\big|_{L=N=1,D=\exp(-\epsilon_s/N_0)} \tag{3.75}$$

and the decoded information sequence bit error probability is upper bounded by

$$P_B < \operatorname{erfc}\sqrt{\frac{2d\epsilon_s}{N_0}}\exp\left(\frac{d\epsilon_s}{N_0}\right)\frac{\partial T(D,L,N)}{\partial N}\bigg|_{L=N=1,D=\exp(-\epsilon_s/N_0)} \tag{3.76}$$

where

$$\operatorname{erfc}(x) = \int_x^\infty \frac{1}{\sqrt{2\pi}}e^{-t^2/2}\,dt \tag{3.77}$$

3.9 Trellis Coded Modulation (TCM)

By the mid 1970's people began to express the opinion that there was little to be gained by error control coding for heavily band limited channels like voice band telephone circuits. The envisioned problem was that coding requires adding redundancy which seemed to require transmitting more channel symbols per second for a fixed information rate which causes bandwidth expansion. Then Ungerboeck and Csajka [61] disclosed a major breakthrough at the 1976 IEEE Information Theory Symposium showing how significant coding gains could be achieved without expanding bandwidth by combining convolutional

coding and QAM modulation. Their method was to add redundancy by expanding the QAM 2D constellation size beyond the 2^K points required for K information bits, partitioning the constellation into a sequence of subsets with increasing minimum squared Euclidean distance between subset points at each partition level, and using the output bits of the convolutional encoder to assign constellation subsets to trellis branches rather than binary encoded bits. This caused no bandwidth expansion at all. Based on capacity curves, they argued that almost all the potential gain could be achieved by simply doubling the constellation size which meant the convolutional encoder should generate just one check bit and be a $(K+1, K)$ encoder. Gains of more than 3 dB could easily be achieved and the method was very quickly included in commercial wireline modems. The method is now know as *trellis coded modulation* (TCM). Calderbank and Sloane [9] and Forney [20, 21] formalized the set partitioning idea by using the language and theory of lattices. They recognized that the constellation partitioning process was equivalent to coset decompositions of lattices and lattice partition chains.

The block diagram of a typical systematic trellis coded modulation system is shown in Figure 3.10. The serial input data stream is divided into the $K+U$ streams $X_1(D), \ldots, X_{K+U}(D)$. The K input streams $X_1(D), \ldots, X_K(D)$ are applied to a $(K+1, K)$ systematic convolutional encoder to generate the single check stream $Y_0(D)$. The $K+1$ signals consisting of the check stream and first K information bit streams are used to select one of 2^{K+1} cosets of a lattice partition Λ_c/Λ_c' at each time instant. The actual constellation points typically belong to a translation of the coding lattice Λ_c. For example, the coding lattice might be $\mathbf{Z}^2$ with the constellation points selected from $\mathbf{Z}^2 + (0.5, 0.5)$. The coding sublattice Λ_c' must be selected so that the partition Λ/Λ_c' has order 2^{K+1}. The remaining U input bits streams $X_{K+1}(D), \ldots, X_U(D)$ select one of 2^U points from the designated coset of the coding sublattice Λ_c' at each time instant. These bits are often called the *uncoded bits*. Details of trellis coding for V.34 modems are presented in Chapters 9 and 10. The V.34 codes use 4-dimensional constellations.

According to Ungerboeck's set partitioning method, the encoder output bits $[y_0, y_1, \ldots, y_K]$ correspond to the binary coset label $[a_0, a_1, \ldots, a_K]$ discussed in Section 1.4. (Notice that the order of the a's has been reversed here.) The bit $y_0(n)$ determines which of the two partitions of Λ_c is chosen at the top level, $y_2(n)$ determines which of the two partitions are chosen at the next level, etc. At each level the minimum distance between points within a coset (or subset, as they are called by Ungerboeck) increases. The minimum distance at the top level, that is, in the lattice Λ_c, will be designated by d_0, at the next level as d_1, etc.

The error probability for a trellis code can be estimated at high signal-to-noise ratios from the minimum Euclidean distance between constellation sequences

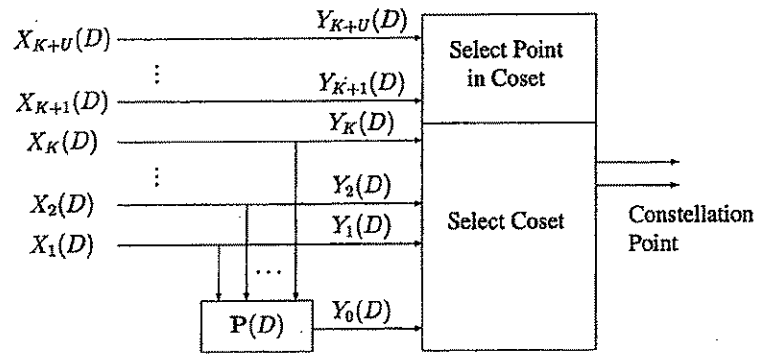


Figure 3.10. Block Diagram for a Systematic Trellis Encoder

selected by the encoder along trellis paths that diverge for the first time at a node and later remerge forever. This minimum distance will be called $d_{\min}$.

EXAMPLE 3.9 Ungerboeck's Non-Systematic 4-State Trellis Code

Ungerboeck's 4-state trellis code is based on the partition chain $\mathbf{Z}^2/R\mathbf{Z}^2/2\mathbf{Z}^2$. The partition $\mathbf{Z}^2/2\mathbf{Z}^2$ has order 4 and each sub-partition has order 2. Consider the non-systematic version of Ungerboeck's 4-state encoder shown in Figure 3.4. Representatives of the four cosets of $2\mathbf{Z}^2$ can be expressed as

$$\mathbf{C}(y_1, y_2) = [y_1\ y_2]\begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix} = \mathbf{YQ} \tag{3.78}$$

The input-output relationship for the Ungerboeck 4-state non-systematic code is

$$\mathbf{Y}(D) = [Y_1(D)\ Y_2(D)] = \mathbf{X}_1(D)[D\ \ 1+D^2] \tag{3.79}$$

Thus the coset representatives can be computed as

$$\begin{aligned} \mathbf{C}(\mathbf{Y}(D)) &= \mathbf{X}_1(D)[D\ \ 1+D^2]\begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix} \\ &= \mathbf{X}_1(D)[1+D^2\ \ 1+D+D^2] \end{aligned} \tag{3.80}$$

So, an equivalent form of Ungerboeck's encoder whose output is the coset representatives has the generator matrix

$$\mathbf{G}_r(D) = [1+D^2\ \ 1+D+D^2] \tag{3.81}$$

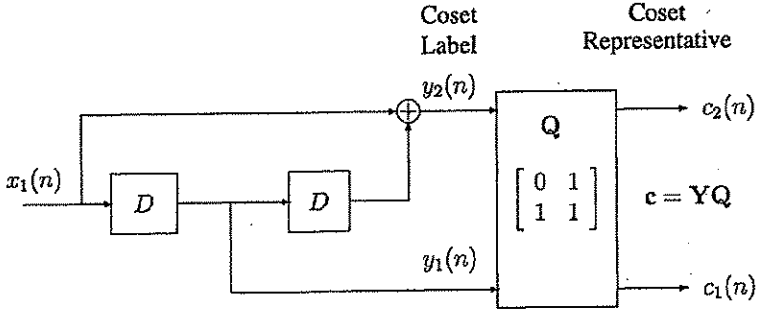
This is the form used in Forney's paper on trellis shaping [23]. A check matrix for the code is

$$\mathbf{H}_r(D) = [1+D+D^2\ \ 1+D^2] \tag{3.82}$$

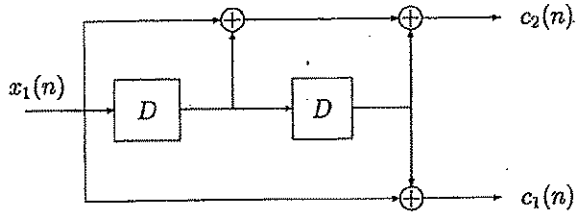
and an inverse check matrix is

$$(\mathbf{H}_r^t)^{-1} = [1/(1 + D + D^2) \;\; 0] \tag{3.83}$$

A block diagram for this encoder is shown in Figure 3.11.



(a) Original Encoder Followed by Transformation



(b) Combined Encoder and Transformation

Figure 3.11. Transformation of Ungerboeck's 4-State Code into Forney's Form

The syndrome for the received sequence $\mathbf{R}(D) = [r_1(D) \;\; r_2(D)]$ is

$$S(D) = r_1(D)(1 + D + D^2) + r_2(D)(1 + D^2) \tag{3.84}$$

A block diagram for this syndrome former is shown in Figure 3.12. A block diagram of the inverse syndrome former is shown in Figure 3.13.

The lattice partition tree for this code is illustrated in Figure 3.14.This corresponds to the partition equation

$$\mathbf{Z}^2 = 2\mathbf{Z}^2 + y_1[0 \;\; 1] + y_1[1 \;\; 1] \tag{3.85}$$

Notice that

$$R\mathbf{Z}^2 = 2\mathbf{Z}^2 \cup \{2\mathbf{Z}^2 + [1 \;\; 1]\} \tag{3.86}$$

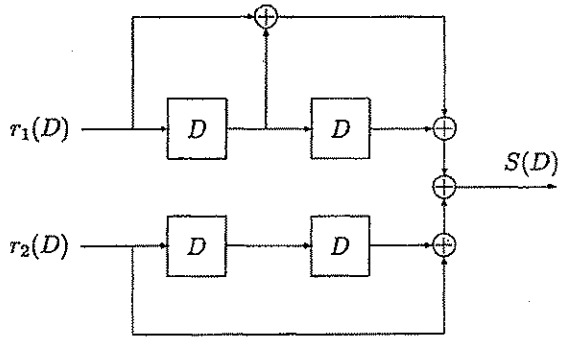


Figure 3.12. Syndrome Former

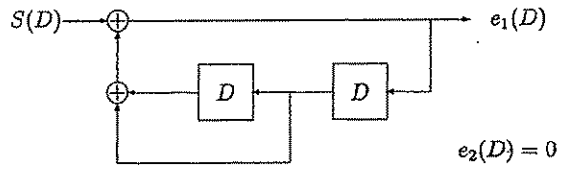


Figure 3.13. Inverse Syndrome Former

and

$$\mathbf{Z}^2 = R\mathbf{Z}^2 \cup \{R\mathbf{Z}^2 + [0 \;\; 1]\} \tag{3.87}$$

The minimum squared Euclidean distance (MSED) for this code will now be determined. The trellis diagram for this code is repeated in Figure 3.15 with the all zeros path and the minimum distance path that diverges from all zeros and remerges shown by wider lines. Notice that the two branches that diverge from any node always have the same y_1 bit. Either $y_1 = 0$ for both or $y_1 = 1$ for both. Also, the two branches that converge on each node have the same y_1 value. This is shown explicitly in the trellis diagram but also becomes evident from the encoder block diagram. The current y_1 value taken from between the two delay elements does not depend on the current input bit, so paths diverging from the current state have the same y_1. The value of $y_1(n-1)$ for paths converging on a state at time n is the value of the state variable at the output of the delay element on the right of the encoder diagram. Therefore, the two paths converging on a state must have had the same y_1 at the previous time instant. The values of y_2 for branches converging on or diverging from a state are always different.

Whenever y_1 is the same for two branches, the constellation points must be from the same first level partition, that is, either from $R\mathbf{Z}^2$ for $y_1 = 0$ or

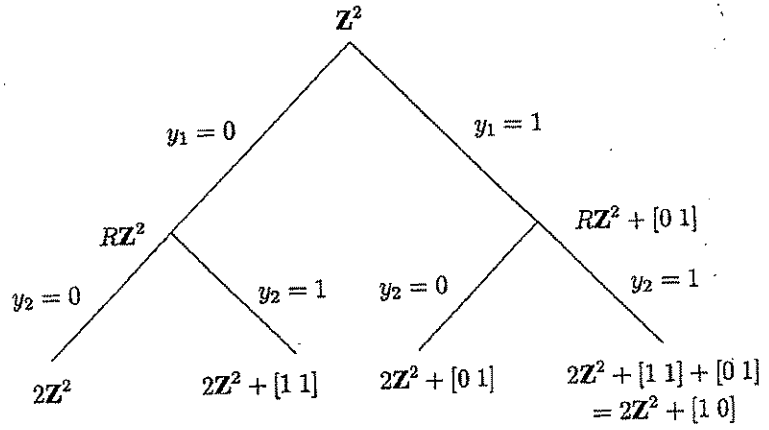


Figure 3.14. The Partition Tree

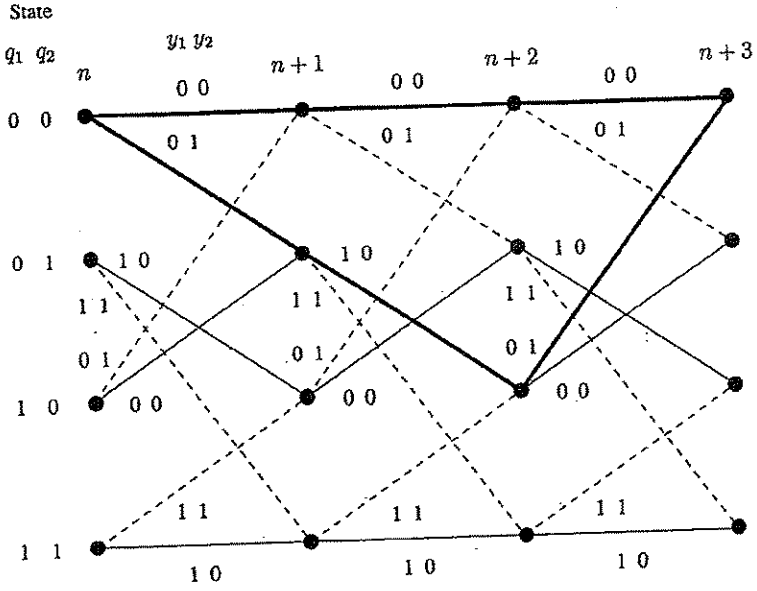


Figure 3.15. Trellis Diagram Showing a Pair of Minimum Distance Paths

from $R\mathbf{Z}^2 + [0\ \ 1]$ for $y_1 = 1$. In either case, the squared distance between the two points must be no less than the minimum in $R\mathbf{Z}^2$ which is $d_1^2 = 2$. The two darkened paths in the trellis diagram shown in Figure 3.15 extend over three branches. The segments that diverge from state 00 or remerge with state 00 must each have minimum squared distance 2. The branches of the middle segment have $[y_1\ \ y_2] = [0\ \ 0]$ or $[1\ \ 0]$. Thus, the constellation points on the upper branch belong to $R\mathbf{Z}^2$ and on the lower branch to $R\mathbf{Z}^2 + [0\ \ 1]$. Since the union of these two subsets is $\mathbf{Z}^2$, the minimum squared distance between them is the minimum distance in $\mathbf{Z}^2$ which is $d_0^2 = 1$. Therefore, the minimum squared distance between the two paths is

$$d_f^2 = 2d_1^2 + d_0^2 = 5 \tag{3.88}$$

The distance between paths that actually diverge for more than one branch before remerging is called the *minimum free distance*.

Each branch in the trellis represents a coset selection. The particular point in the coset is selected by the uncoded input bits. Thus, each branch really represents a set of parallel transitions which are paths that diverge and remerge one node later. The minimum distance, $d_p = 2$, within each coset of the coding sublattice $2\mathbf{Z}^2$ has to be considered when finding the minimum distance for the trellis code. The minimum squared Euclidean distance for the trellis code is

$$d_{\min}^2 = \min\{d_p^2, d_f^2\} = 4 \tag{3.89}$$

Ungerboeck has given a set of rules for selecting good trellis codes. One of his rules is that paths diverging from a state should have the same first level partition bit (y_0 in Figure 3.10 or y_1 in the previous example) and, similarly, all paths converging on a state should have the same first level partition bit. This ensures a minimum squared distance along diverging and remerging paths of at least $2d_1^2$. For systematic $(K+1, K)$ codes, this can be insured by having no input bits add into the input of the delay element on the left or the output of the delay element on the right. This is clearly the case for the systematic form of the Ungerboeck 4-state code shown in Figure 3.6.

3.10 Brief Review of the Viterbi Decoding Algorithm

The Viterbi decoding algorithm is a method for finding the maximum likelihood estimate of a transmitted trellis sequence given the corresponding received sequence. In this section we will assume that 2D QAM constellation points are assigned to trellis branches and that they are transmitted over an additive white Gaussian noise channel. The constellation points will be considered to be complex numbers. The method can be easily adapted to other schemes like transmission of the binary encoder outputs over a binary symmetric channel or

by BPSK over an AWGN channel. Suppose the sequence of 2D constellation points transmitted along a trellis path is $\mathbf{u}(n)$ for $n = 0, \ldots, L-1$ and the received sequence is

$$\mathbf{r}(n) = \mathbf{u}(n) + \mathbf{v}(n) \quad \text{for } n = 0, \ldots, L-1 \tag{3.90}$$

where $\mathbf{v}(n)$ is a white, complex, zero-mean Gaussian noise sequence with variance $\sigma^2 = \mathcal{E}|\mathbf{v}(n)|^2$. We will assume that the encoder starts in state 0 at time $n = 0$ and is forced to end in state 0 at time L by appending an appropriate string of information bits to the input data stream. For example, if the encoder is realized using the type 1 direct form shown in Figure 3.2 with M delay elements, the last M inputs should equal the feedback signal to the adder on the left to clear the delay line to the zero state.

The conditional probability density function for the received sequence given a possible trellis sequence is

$$f(\mathbf{r}_0, \ldots, \mathbf{r}_{L-1} \,|\, \mathbf{u}_0, \ldots \mathbf{u}_{L-1}) = \frac{1}{(\sigma^2\pi)^{L/2}} e^{-\sum_{n=0}^{L-1}|\mathbf{r}(n)-\mathbf{u}(n)|^2/\sigma^2} \tag{3.91}$$

A maximum likelihood receiver selects the sequence $\mathbf{u}(n)$ that maximizes this pdf which is also called the *likelihood function*. In this case, the likelihood function is maximized if $\mathbf{u}(n)$ is chosen to minimize

$$\Gamma_0(L) = \sum_{n=0}^{L-1} |\mathbf{r}(n) - \mathbf{u}(n)|^2 \tag{3.92}$$

The function $\Gamma_0(L)$ is the squared Euclidean distance between the received sequence $\mathbf{r}(n)$ and hypothesized transmitted trellis sequence $\mathbf{u}(n)$. The subscript 0 indicates that the encoder is forced to end in state 0 at time L. $\Gamma_0(L)$ is also called the *cumulative sequence metric*. An individual term, $|\mathbf{r}(n) - \mathbf{u}(n)|^2$, in the sum is called a *branch metric*. Stated in another way, a maximum likelihood decoder should select the trellis sequence that is closest to the received sequence in Euclidean distance.

The maximum likelihood path through the trellis can be found by iteratively progressing from the beginning to the end of the trellis. Let the encoder have S states labeled $0, \ldots, S-1$, and let the state at time n be $s(n)$. The trellis is forced to have the boundary conditions $s(0) = s(L) = 0$. Suppose that somehow we know the best trellis paths and corresponding cumulative metrics to each of the states at time n. A trellis path consists of the sequence of states traversed and the constellation points associated with the branches connecting the states. By "best" we mean the path from state 0 at time 0 to the state at time n that minimizes the sum of the branch metrics along the path. Let the sequence of constellation points along the best path to state i at time n be $\mathbf{u}^*(k; s(n) = i)$ for $k = 0, \ldots, n$ and let the cumulative metric for the best path to state i at time n be

$$\Gamma_i^*(n) = \sum_{k=0}^{n-1} |\mathbf{r}(k) - \mathbf{u}^*(k; s(n) = i)|^2 \tag{3.93}$$

The best paths are called the *survivors* to the states. Any path from the beginning to the end of the trellis that passes through a particular state at time n must follow the survivor to that state up to time n since the survivor has the minimum metric up to that state and the branch metrics for paths leading out of the state are non-negative and get added to the cumulative metric of the survivor. At time n we can form a data record consisting of (1) an S-dimensional array where the ith element is the cumulative metric of the survivor to state i, (2) an S-dimensional array where the ith element is the state at time $n-1$ where the survivor to state i at time n came from, and (3) an S dimensional array where the ith element contains the subset point selected for the surviving branch connecting to state i. Item (3) must be saved because from knowledge of two successive states in the trellis only the constellation subset can be determined. The point within the subset is selected by the uncoded bits and this information must be saved. For a traditional binary convolutional code, there are no parallel transitions and item (3) is not required.

Next the survivors and their cumulative metrics to the states at time $n+1$ can be found using the known results at time n. When the convolutional encoder has K inputs, 2^K paths diverge from and converge on each state. The 2^K candidates for the survivor to a state i at time $n+1$ must be extensions of the survivors to states at time n that have branches connecting to state i at time $n+1$. Let $\mathcal{C}_i$ be the set of 2^K states converging on state i. A branch converging on state i from a state $j \in \mathcal{C}_i$ has a subset of 2^U constellation points assigned to it which we will call $\mathcal{U}_{j,i}$. Since the cumulative metric at time $n+1$ is the sum of the cumulative metric of the survivor to state j at time n and the branch metric for the branch connecting to state i at time $n+1$, the best choice for the constellation point along this path must be the point $\mathbf{u}_{j,i}^*(n) \in \mathcal{U}_{j,i}$ that is closest to the received point $\mathbf{r}(n)$ in Euclidean distance. The process of selecting this subset point is often called *slicing* the received point to the subset. After performing slicing, the 2^K cumulative metrics for the survivor candidates to state i at time $n+1$ are

$$\Gamma_i(n+1) = |\mathbf{r}(n) - \mathbf{u}_{j,i}^*(n)|^2 + \Gamma_j^*(n) \quad \text{for } j \in \mathcal{C}_i \tag{3.94}$$

The Viterbi decoder then selects the $j \in \mathcal{C}_i$ which gives the minimum $\Gamma_i(n+1)$ and records this new survivor metric and optimum previous state in the decoding data record for time $n+1$. The cumulative metric for the survivor to state i at time $n+1$ is

$$\Gamma_i^*(n+1) = \min_{j \in \mathcal{C}_i} \left\{ |\mathbf{r}(n) - \mathbf{u}_{j,i}^*(n)|^2 + \Gamma_j^*(n) \right\} \tag{3.95}$$

This survivor selection must be performed for each state at time $n + 1$.

The process of survivor selection can be started at time $n = 0$ when the encoder is known to be in state 0 and extended stage-by-stage into the trellis until time L when the encoder is again known to be in state 0. At time L all trellis paths converge to state 0 and a single survivor to this state is selected. This survivor to state 0 at time L is the maximum likelihood estimate of the transmitted sequence. Since the stored data record for each state at a given time contains a pointer back to the previous best state connecting to it, the path can be *traced back* from the survivor to state 0 at time L to the beginning of the trellis to find this maximum likelihood path. This use of pointers eliminates the computational burden of swapping entire paths from the beginning of the trellis up to the current depth as new survivors are found at each iteration.

Finally, let us consider some practical matters. In many cases, the trellis length L is essentially unlimited. For example, the trellis codes in V.32 and V.34 modems run from the time the modems initially connect until the call is terminated so L is essentially infinite. Clearly, the storage for the decoding data records at each time must be limited. The solution is to limit the storage to I stages back from the current time where I is several times the constraint length of the encoder. That is, the records are stored from time n back to time $n - I$, typically, in a circular buffer. Simulations have shown that as paths are traced back from the states at time n, they all merge into a common tail when there is not excessive noise. Little is to be gained by saving records back beyond the point where the paths merge into the common tail. No good design equations exist for choosing I. The best approach seems to be by simulations. As I is increased, the coding gain initially increases rapidly but then exponentially converges to a maximum value. The decoder at each time instant n finds the state with minimum cumulative metric and traces the path from that state back I stages to the end of the data record storage and outputs the constellation point stored there. A decoder of this type introduces a decoding delay of I trellis branches. It may seem that the trace back can be started from any state since it is assumed all the paths merge into a common tail. However, simulations have shown that when I is not very large, starting the trace back from the state with minimum metric can improve performance by a couple of tenths of a dB.

Another problem is that the cumulative metrics continue to grow and can eventually overflow the computer word length. In making metric comparisons to find the surviving paths, only the relative sizes of the metrics are important. Therefore, the overflow problem can be solved by periodically resetting the metrics by subtracting the minimum metric from all the metrics at that stage. In terms of implementation logic, it is easiest to do the resetting at each iteration.

3.11 The Fundamental Coding Gain of a Trellis Code

Suppose a trellis code uses an (n, k) convolutional code we will designate by C to select a sequence of cosets from the lattice partition Λ/Λ' where Λ and Λ' are N-dimensional lattices. The order of the partition must be $|\Lambda/\Lambda'| = 2^n$. The expression $\mathbb{C}(\Lambda/\Lambda'; C)$ will be used to denote this trellis code. The redundancy of the convolutional code is defined as $r(C) = n - k$ and the normalized redundancy as $\rho(C) = 2r(C)/N$. The redundancy of the lattice Λ was defined in Section 1.4 as $r(\Lambda) = |\mathbf{Z}^N/\Lambda|$ and redundancy normalized to two dimensions as $\rho(\Lambda) = 2r(\Lambda)/N$. The total redundancy of the trellis code is

$$r(\mathbb{C}) = r(C) + r(\Lambda) \tag{3.96}$$

The total redundancy of the trellis code normalized to two dimensions is

$$\rho(\mathbb{C}) = \rho(C) + \rho(\Lambda) \tag{3.97}$$

The coding gain of a trellis code has been defined in a variety of ways. Forney [20] shows how to express the total coding gain as the product of a factor $\gamma_s(\mathbb{C})$ called the shaping gain and a factor $\gamma(\mathbb{C})$ called the *fundamental coding gain*. The shaping gain is a measure of the power reduction resulting from using a constellation more spherically shaped than an N-cube and is discussed in Section 2.5. The shaping gain depends on the constellation boundary and is independent of the convolutional code and lattice partition. It is a small fraction of the total coding gain and is upper bounded by the 1.53 dB shaping gain of an infinite dimensional sphere. Forney explains how to define the fundamental volume of a trellis code and based on this definition defines the fundamental coding gain of the trellis code as

$$\gamma(\mathbb{C}) = 2^{-\rho(\mathbb{C})} d^2_{\min}(\mathbb{C}) \tag{3.98}$$

where $d^2_{\min}(\mathbb{C})$ is the minimum squared Euclidean distance for $\mathbb{C}$. The fundamental coding gain is independent of the constellation shape and is much larger than the shaping gain for good codes. The fundamental coding gain is a good measure of code performance when the constellation has many points.

EXAMPLE 3.10 Fundamental Coding Gain for Ungerboeck 4-State Code

For this code, the convolutional code normalized redundancy is $\rho(C) = 1$ and the normalized redundancy of the coding lattice is $\rho(\mathbf{Z}^2) = 0$, so the total normalized redundancy is $\rho(\mathbb{C}) = 1$. We have seen that $d^2_{\min}(\mathbb{C}) = 4$. Therefore, the fundamental coding gain is

$$\gamma(\mathbb{C}) = 4/2 = 2 \text{ or } 3.01 \text{ dB} \tag{3.99}$$

■

Chapter 4

TRELLIS SHAPING

Early in the ITU-T V.34 committee deliberations, G. David Forney of Motorola Information Systems proposed including a constellation shaping method called *trellis shaping* in the recommendation. Forney initially presented his ideas at the Information Theory Workshop at Cornell University in June 1989 and an enhanced version of his paper was subsequently published in the *IEEE Transactions on Information Theory* [23]. The concept of constellation design to achieve shaping gain was discussed in Chapter 2 and results for finite dimensional constellations selected from the interior of an N-sphere were presented. A practical method for constellation shaping called shell mapping using finite dimensional constellations is presented in Chapter 8. Shell mapping approximates the ideal N-sphere mapping but with peak signal constraints and reduced complexity. The trellis shaping techniques presented in this chapter use infinite dimensional constellations consisting of infinite duration sequences of 2D symbols. The transmitted sequences are selected from an equivalence class of possible sequences to minimize the transmitted signal power. The resulting 2D distributions of transmitted points are not uniform, with points of low power near the origin more likely than high power points far from the origin just as for N-sphere and shell mapping. As with finite dimensional shaping, the 2D constellation size must be expanded relative to the size of unshaped constellations to achieve shaping gain.

Two methods for trellis shaping are presented in this chapter. The first method, trellis shaping based on lattice partitions, grew from work on finite dimensional shaping using the Voronoi regions of N-dimensional lattices [22]. The shaped constellations can be viewed as belonging to the infinite dimensional Voronoi region of a trellis code. Later, the second method, trellis shaping on regions, was created by Forney when he realized the work of Calderbank and Ozarow [8] on finite dimensional shaping on regions could be generalized to

infinite sequences. Both methods yield comparable shaping gains for equivalent complexity.

Trellis shaping was soon generalized to include non-linear precoding for channel equalization [18] and this is discussed in Chapter 6 Trellis Precoding. Although trellis shaping and trellis precoding can achieve significant shaping gains, the V.34 study committee considered them to be too complex computationally and chose shell mapping with the LTF precoder instead. Trellis shaping and trellis precoding are presented in this book because they are conceptually very interesting techniques and for historical completeness.

4.1 Trellis Shaping Based on Lattice Partitions

A block diagram of an encoder for trellis shaping based on lattice partitions is shown in Figure 4.1. To add generality, the figure indicates that transmitted channel symbols are taken from an N-dimensional constellation. In most actual modems, an $N = 2$ or, possibly, 4-dimensional constellation would be used. Each N-dimensional symbol interval, the shaping encoder takes in k_c bits to form $\mathbf{x}$, n_u bits to form $\mathbf{w}$, and r_s bits to form $\mathbf{s}$ for a total of $k_c + n_u + r_s$ bits per N-dimensional symbol. The input $\mathbf{x}$ is applied to a channel trellis encoder whose output is the binary n_c-tuple $\mathbf{y}$. The vector $\mathbf{w}$ corresponds to the uncoded input bits for an ordinary channel trellis encoder. The *shaping bits* $\mathbf{s}$ pass through an inverse syndrome former for a shaping trellis code to give the binary n_s-tuple $\mathbf{t}$ which is a shaping coset representative. With channel coding but no shaping, a constellation with $M_1 = 2^{n_c+n_u+r_s}$ points is required. The shaping process increases the number of shaping bits from r_s to $n_s = r_s + k_s$ bits so the shaped constellation requires $M_2 = 2^{n_c+n_u+n_s}$ points and the constellation expansion ratio normalized to two dimensions is

$$\mathrm{CER}_s = M_2/M_1 = 2^{2k_s/N} \tag{4.1}$$

A block diagram of the receiver is shown in Figure 4.2. The large majority of the computational effort of the receiver is devoted to the Viterbi decoder for the channel trellis code. The different components of the trellis shaping encoder and receiver are described in more detail in the following sections.

4.1.1 The Trellis Shaping Encoder

The components of the trellis shaping encoder shown in Figure 4.1 will be described in detail in this section.

4.1.1.1 The Channel Trellis Code

The channel trellis code is a traditional trellis code used to achieve coding gain. The collection of possible trellis sequences will be denoted by $\mathbb{C}_c = \mathbb{C}(\Lambda_c/\Lambda_c'; C_c)$. C_c represents a binary (n_c, k_c) convolutional code with

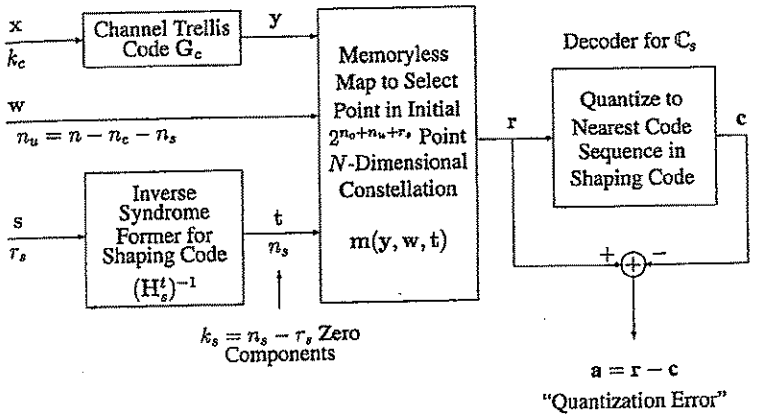


Figure 4.1. Encoder for Trellis Shaping Based on Lattice Partitions

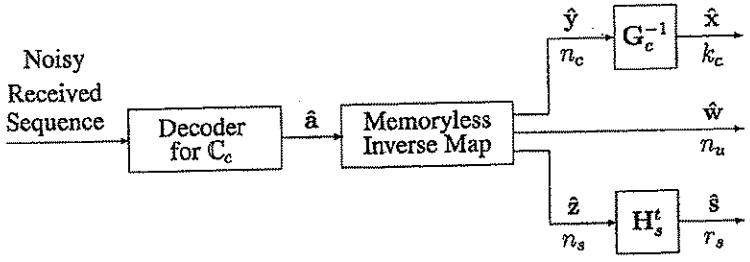


Figure 4.2. Receiver for Trellis Shaped Constellation

generator matrix $\mathbf{G}_c$. Its input is the k_c-tuple $\mathbf{x}$ and its output is the n_c-tuple $\mathbf{y}$. Both Λ_c and Λ_c' are binary lattices with Λ_c' a sublattice of Λ_c. The order of the lattice partition Λ_c/Λ_c' must be 2^{n_c}. The actual transmitted constellation points are selected from the translated lattice $\Lambda_c + \mathbf{d}$. A common example is $\Lambda_c = 2\mathbf{Z}^2$ and $\mathbf{d} = (1,1)$.

Let $[\Lambda_c/\Lambda_c']$ designate a set of 2^{n_c} coset representatives for the lattice partition Λ_c/Λ_c'. The output $\mathbf{y}$ of the convolutional code C_c selects cosets of Λ_c' through a mapping or function we will call $\mathbf{m}_c(\cdot)$ whose domain is the set of all possible binary n_c-tuples and whose range is the set of coset representatives. The coset representatives are $\mathbf{m}_c(\mathbf{y})$ and are points in the lattice Λ_c. Thus the cosets are $\Lambda_c' + \mathbf{m}_c(\mathbf{y})$ and the transmitted points have the form

$$\lambda_c' + \mathbf{m}_c(\mathbf{y}) + \mathbf{d} \tag{4.2}$$

where $\lambda_c' \in \Lambda_c'$ and $\mathbf{d}$ is a point in a fundamental region $\mathcal{R}(\Lambda_c)$ of Λ_c.

4.1.1.2 The Uncoded Bits w

As in the traditional trellis codes, the uncoded bits $\mathbf{w}$ select points from the coset of Λ_c' selected by the output $\mathbf{y}$ of the convolutional encoder C_c. Let the shaping lattice Λ_s be a binary sublattice of Λ_c' with $|\Lambda_c'/\Lambda_s| = 2^{n_u}$ and let $[\Lambda_c'/\Lambda_s]$ be a set of 2^{n_u} coset representatives for the partition Λ_c'/Λ_s. The binary n_u-tuple $\mathbf{w}$ selects the cosets of Λ_s in Λ_c' through a mapping function $\mathbf{m}_u(\cdot)$ whose domain is the set of all binary n_u-tuples and whose range is the set of coset representatives $[\Lambda_c'/\Lambda_s]$. Each coset representative is a point $\mathbf{m}_u(\mathbf{w})$ in Λ_c' and the cosets are $\Lambda_s + \mathbf{m}_u(\mathbf{w})$ and belong to Λ_c'. Since each point λ_c' in Λ_c' can be represented as $\lambda_s + \mathbf{m}_u(\mathbf{w})$ for some $\mathbf{w}$ and $\lambda_s \in \Lambda_s$, the transmitted points can be expressed as

$$\lambda_c' + \mathbf{m}_c(\mathbf{y}) + \mathbf{d} = \underbrace{[\lambda_s + \mathbf{m}_u(\mathbf{w})]}_{\in \Lambda_c'} + \mathbf{m}_c(\mathbf{y}) + \mathbf{d} \tag{4.3}$$

Notice that these points still belong to the same translated coset of Λ_c' selected by the output $\mathbf{y}$ of the channel convolutional code C_c.

4.1.1.3 The Trellis Shaping Code and Inverse Syndrome Former

Constellation shaping is performed with the aid of a *trellis shaping code* $\mathbb{C}_s = \mathbb{C}(\Lambda_s/\Lambda_s'; C_s)$ where C_s is a binary (n_s, k_s) convolutional code, Λ_s' is a binary sublattice of Λ_s, and $|\Lambda_s/\Lambda_s'| = 2^{n_s}$. The lattice Λ_s is used untranslated. At this point, we have a lattice partition chain $\Lambda_c/\Lambda_c'/\Lambda_s/\Lambda_s'$ with $|\Lambda_c/\Lambda_s'| = 2^{n_c+n_u+n_s}$.

All sequences $\mathbf{c}(D)$ in the convolutional code C_s satisfy the parity check equation

$$\mathbf{c}(D)_{1\times n_s}\mathbf{H}_s^t(D)_{n_s\times(n_s-k_s)} = \mathbf{0}_{1\times(n_s-k_s)} \tag{4.4}$$

and, conversely, any sequence that satisfies the parity check equation is a code sequence. The matrix $\mathbf{H}_s(D)$ is called the parity check matrix. It can be shown that a feedback free parity check matrix always exists. Let $\mathbf{y}(D) = \mathbf{c}(D) + \mathbf{t}(D)$ where $\mathbf{c}(D)$ is a code sequence and $\mathbf{t}(D)$ is not a code sequence. The *syndrome* for $\mathbf{y}(D)$ is defined to be

$$\mathbf{s}(D) = \mathbf{y}(D)\mathbf{H}_s^t(D) = [\mathbf{c}(D) + \mathbf{t}(D)]\mathbf{H}_s^t(D) = \mathbf{t}(D)\mathbf{H}_s^t(D) \tag{4.5}$$

Notice that the syndrome does not depend on the chosen code sequence $\mathbf{c}(D)$ at all. It depends only on the "error sequence" $\mathbf{t}(D)$. The set of sequences

$$C_s + \mathbf{t}(D) \triangleq \{\mathbf{c}(D) + \mathbf{t}(D) \,|\, \mathbf{c}(D) \in C_s\} \tag{4.6}$$

is an equivalence class of sequences in the sense that each member has the same syndrome. The error sequence $\mathbf{t}(D)$ is a representative for this equivalence class. The representative is not unique since any member of the equivalence class can be used as its representative. The equivalence class can also be viewed as a coset of the set of code sequences in the space of infinite dimensional binary sequences.

In Section 3.6, a method for finding an equivalence class representative from a syndrome is discussed. It is shown how to construct an inverse check matrix or inverse syndrome former $(\mathbf{H}_s^t)^{-1}$ such that

$$\mathbf{t}(D) = \mathbf{s}(D)(\mathbf{H}_s^t(D))^{-1} \tag{4.7}$$

is an equivalence class representative with $k_s = n_s - r_s$ zero components and an example for Forney's form of the Ungerboeck 4-state code is given in Example 3.9. The inverse syndrome former may have feedback but this is not important because it is in the transmitter where perfect data is available and no error propagation will occur. The basic idea behind trellis shaping is to choose the sequence in the equivalence class represented by $\mathbf{t}(D)$ that minimizes the average transmitted constellation point energy.

The binary n_s-tuple $\mathbf{t}$ selects one of the 2^{n_s} coset representatives from $[\Lambda_s/\Lambda_s']$ through a function $\mathbf{m}_s(\cdot)$ whose domain is the set of binary n_s-tuples and range is the set of these coset representatives. Therefore, the cosets of Λ_s' in Λ_s are $\Lambda_s' + \mathbf{m}_s(\mathbf{t})$ with $\mathbf{m}_s(\mathbf{t}) \in \Lambda_s$ and any point λ_s in Λ_s has the form $\lambda_s = \lambda_s' + \mathbf{m}_s(\mathbf{t})$ with $\lambda_s' \in \Lambda_s'$. Substituting this result into (4.3), we find that the initial points $\mathbf{r}$ selected by the memoryless map $\mathbf{m}(\mathbf{y}, \mathbf{w}, \mathbf{t})$ have the form

$$\mathbf{r} = \underbrace{\underbrace{\lambda_s' + \mathbf{m}_s(\mathbf{t})}_{\in \Lambda_s} + \mathbf{m}_u(\mathbf{w}) + \mathbf{m}_c(\mathbf{y})}_{\in \Lambda_c'} + \mathbf{d} \tag{4.8}$$

Notice that $\mathbf{r}$ is still in the translated coset of Λ_c' selected by $\mathbf{y}$. Also $\lambda_s' + \mathbf{m}_s(\mathbf{t}) + \mathbf{m}_u(\mathbf{w})$ is in the same coset of Λ_s in Λ_c' selected by $\mathbf{w}$ independent

of the value of $\mathbf{t}$. The significance of this property is that $\mathbf{y}$ and $\mathbf{w}$ will be unaffected by the shaping operation.

The trellis shaping code used in a system employing trellis shaping based on lattice partitions must have a linear mapping from coset labels to coset representatives, that is,

$$\mathbf{m}_s(\mathbf{t} \oplus \mathbf{v}) = \mathbf{m}_s(\mathbf{t}) \pm \mathbf{m}_s(\mathbf{v}) \text{ modulo } \Lambda_s' \tag{4.9}$$

We saw an example of a linear mapping with the Ungerboeck 4-state code with $\Lambda_s = \mathbf{Z}^2$ and $\Lambda_s' = 2\mathbf{Z}^2$ in Example 3.9.

4.1.1.4 Forming the Transmitter Output from $\mathbf{r} = \mathbf{m}(\mathbf{y}, \mathbf{w}, \mathbf{t})$

The final step in the transmitter is to "quantize" the memoryless map output sequence $\mathbf{r} = \mathbf{m}(\mathbf{y}, \mathbf{w}, \mathbf{t})$ to the sequence in the shaping trellis code $\mathbb{C}_s$ that is closest in squared Euclidean distance and transmit the quantization error sequence. According to (4.8) the memoryless map output has the form

$$\mathbf{r} = \lambda_c' + \mathbf{m}_c(\mathbf{y}) + \mathbf{d} \tag{4.10}$$

with $\lambda_c' \in \Lambda_c'$ for all $\mathbf{t}$ and $\mathbf{w}$. This sequence is applied to a decoder for the trellis code $\mathbb{C}_s$. The values of $\mathbf{r}$ belong to the finely spaced translated lattice $\Lambda_s + \mathbf{d}$ while the values of a trellis sequence from the shaping code belong to the coarser lattice Λ_s which is a sublattice of Λ_c. The output of the decoder for $\mathbb{C}_s$ is the code sequence in $\mathbb{C}_s$ closest to $\mathbf{r}$ in Euclidean distance. In other words, the decoder quantizes the sequence $\mathbf{r}$ to a shaping code sequence. This decoding can be performed by a Viterbi decoder. Suppose the decoder output is the lattice point sequence $\mathbf{c}(n)$. At each symbol instant n, the symbols $\mathbf{c}(n)$ belong to Λ_s and also to Λ_c' since Λ_s is a sublattice of Λ_c'. Then the final step in the decoder is to form and transmit the quantization error

$$\mathbf{a} = \mathbf{r} - \mathbf{c} = (\lambda_c' - \mathbf{c}) + \mathbf{m}_c(\mathbf{y}) + \mathbf{d} = \hat{\lambda}_c' + \mathbf{m}_c(\mathbf{y}) + \mathbf{d} \tag{4.11}$$

with $\hat{\lambda}_c' = \lambda_c' - \mathbf{c} \in \Lambda_c'$. So $\mathbf{a}$ is in the translated coset of Λ_c' selected by $\mathbf{y}$ and the channel trellis code bits are unaffected by the shaping operation. Furthermore, the memoryless map output points have the form

$$\mathbf{r} = \lambda_s + \mathbf{m}_u(\mathbf{w}) + \mathbf{m}_c(\mathbf{y}) + \mathbf{d} \text{ with } \lambda_s \in \Lambda_s \text{ and } \mathbf{m}_u(\mathbf{w}) \in \Lambda_c \tag{4.12}$$

so

$$\begin{aligned} \mathbf{a} &= \mathbf{r} - \mathbf{c} = (\lambda_s - \mathbf{c}) + \mathbf{m}_u(\mathbf{w}) + \mathbf{m}_c(\mathbf{y}) + \mathbf{d} \\ &= \hat{\lambda}_s + \mathbf{m}_u(\mathbf{w}) + \mathbf{m}_c(\mathbf{y}) + \mathbf{d} \end{aligned} \tag{4.13}$$

Thus the transmitted points, $\mathbf{a}$, are in the translated coset of the partition Λ_c/Λ_s selected by $\mathbf{w}$ and $\mathbf{y}$ independent of the shaping operation.

The effect of the shaping operation on $\mathbf{t}$ will now be examined. Let $\mathbf{v}(D)$ represent a codeword in the shaping convolutional code C_s. At each symbol time instant, the shaping encoder output is a binary n_s-tuple. The shaping trellis code sequence has the form

$$\mathbf{c} = \hat{\lambda}_s' + \mathbf{m}_s(\mathbf{v}) \text{ with } \hat{\lambda}_s' \in \Lambda_s' \tag{4.14}$$

Using the representation for the memoryless map output $\mathbf{r}$ given by (4.8), we see that the quantization error has the form

$$\mathbf{a} = \mathbf{r} - \mathbf{c} = (\lambda_s' - \hat{\lambda}_s') - \mathbf{m}_s(\mathbf{v}) + \mathbf{m}_s(\mathbf{t}) + \mathbf{m}_u(\mathbf{w}) + \mathbf{m}_c(\mathbf{y}) + \mathbf{d} \tag{4.15}$$

Notice that $\tilde{\lambda}_s' = \lambda_s' - \hat{\lambda}_s'$ is an element of Λ_s' and since $\mathbf{m}_s(\cdot)$ is a linear mapping

$$\mathbf{a} = \tilde{\lambda}_s' + \mathbf{m}_s(\mathbf{t} \oplus \mathbf{v}) + \mathbf{m}_u(\mathbf{w}) + \mathbf{m}_s(\mathbf{y}) + \mathbf{d} \tag{4.16}$$

Therefore, the shaping operation can be viewed as selecting the minimum Euclidean energy sequence in the equivalence class

$$\{\mathbf{a}\} = \{\Lambda_s' + \mathbf{m}_s(\mathbf{z}) + \mathbf{m}_u(\mathbf{w}) + \mathbf{m}_c(\mathbf{y}) + \mathbf{d} \,|\, \mathbf{z} \in \mathbf{t} \oplus C_s\} \tag{4.17}$$

4.1.2 The Receiver

The first step in the receiver is to make an estimate, $\hat{\mathbf{a}}$, of the transmitted sequence, $\mathbf{a}$. This can be done by using a Viterbi decoder for the channel code $\mathbb{C}_c$. A *memoryless inverse map* is then used to estimate $\mathbf{y}$, $\mathbf{w}$, and $\mathbf{z}$ by

1 selecting $\hat{\mathbf{y}}$ as the label for the coset of Λ_c' that $\hat{\mathbf{a}} - \mathbf{d}$ belongs to,

2 selecting $\hat{\mathbf{w}}$ as the label for the coset of Λ_s that $\hat{\mathbf{a}} - \mathbf{d} - \mathbf{m}_c(\hat{\mathbf{y}})$ belongs to,

3 selecting $\hat{\mathbf{z}} = \hat{\mathbf{t}} \oplus \hat{\mathbf{v}}$ as the label for the coset of Λ_s' that $\hat{\mathbf{a}} - \mathbf{d} - \mathbf{m}_c(\hat{\mathbf{y}}) - \mathbf{m}_u(\hat{\mathbf{w}})$ belongs to.

In the receiver, the transmitter bits $\mathbf{t}$ have been changed to $\mathbf{z} = \mathbf{t} \oplus \mathbf{v}$ where $\mathbf{v}$ is a codeword in the shaping convolutional code C_s if no decoding errors have occurred. The initial input shaping bits, $\mathbf{s}$, can be estimated by forming the syndrome

$$\mathbf{z}\mathbf{H}_s^t = \mathbf{t}\mathbf{H}_s^t \oplus \mathbf{v}\mathbf{H}_s^t = \mathbf{t}\mathbf{H}_s^t = \mathbf{s} \tag{4.18}$$

A feedback free syndrome former can be used so that error propagation is limited.

4.1.3 Selection of a Specific Constellation

The transmitted points, $\mathbf{a}$, are specified by $\mathbf{y}$, $\mathbf{w}$, and $\mathbf{z} = \mathbf{t} \oplus \mathbf{v}$. Thus the transmitted constellation must have

$$M_s = 2^{n_c + n_u + n_s} \tag{4.19}$$

points from $\Lambda_c + \mathbf{d}$. Remember that $|\Lambda_c/\Lambda_s'| = 2^{n_c+n_u+n_s} = M_s$, so every fundamental region $\mathbb{R}(\Lambda_s')$ must contain M_s points from Λ_c, one from each coset of Λ_s' in the partition Λ_c/Λ_s'. Also, $\mathbb{R}(\Lambda_s')$ is the union of M_s fundamental regions $\mathbb{R}(\Lambda_c)$.

The shaping decoder in the transmitter chooses $\mathbf{c}$ to minimize the power in

$$\mathbf{a} = \mathbf{r} - \mathbf{c} \in \{\Lambda_s' + \mathbf{m}_s(\mathbf{z}) + \mathbf{m}_u(\mathbf{w}) + \mathbf{m}_s(\mathbf{y}) + \mathbf{d}\} \tag{4.20}$$

At each symbol instant, $\mathbf{a}(n)$ belongs to to the shifted coset of Λ_s' in Λ_c/Λ_s' with representative point $\mathbf{m}_s(\mathbf{z}) + \mathbf{m}_u(\mathbf{w}) + \mathbf{m}_s(\mathbf{y}) + \mathbf{d}$. The Voronoi region $\bar{\mathbb{R}}_v(\Lambda_s')$ of Λ_s' by definition contains the minimum weight points from each coset of Λ_s' in $\mathbf{R}^N$, the set of all real N-tuples. In other words, the points in $\bar{\mathbb{R}}_v(\Lambda_s')$ are at least as close to the origin $\mathbf{0}$ as to any other point in Λ_s'. The boundary may contain several points from a coset that are equidistant from $\mathbf{0}$ and a nearby neighbor in Λ_s'. Let $\mathbb{R}_v(\Lambda_s')$ be a fundamental region for Λ_s' formed by eliminating the redundant points from $\bar{\mathbb{R}}_v(\Lambda_s')$ on the boundary so that $\bar{\mathbb{R}}_v(\Lambda_s') \supset \mathbb{R}_v(\Lambda_s')$, and $\mathbb{R}_v(\Lambda_s')$ includes one minimum weight point from each coset of Λ_s' in $\mathbf{R}^N$. The fundamental region $\mathbb{R}_v(\Lambda_s')$ must also include M_s points from $\Lambda_c + \mathbf{d}$, one for each coset of Λ_s' in Λ_c. So $\mathbb{R}_v(\Lambda_s')$ contains the minimum weight element in the shifted coset $\Lambda_s' + \mathbf{m}_s(\mathbf{z}) + \mathbf{m}_u(\mathbf{w}) + \mathbf{m}_c(\mathbf{y}) + \mathbf{d}$. Thus at each symbol instant, the transmitted point $\mathbf{a}(n)$ must be in the Voronoi region $\mathbb{R}_v(\Lambda_s')$ of Λ_s'. In other words, the final constellation should consist of $2^{n_c+n_u+n_s}$ points from $\Lambda_c + \mathbf{d}$ which are in $\mathbb{R}_v(\Lambda_s')$, that is, the set of points $\{\Lambda_c + \mathbf{d}\} \cap \mathbb{R}_v(\Lambda_s')$. The transmitted point is the unique point

$$\mathbf{a} = \{\Lambda_s' + \mathbf{m}_s(\mathbf{z}) + \mathbf{m}_u(\mathbf{w}) + \mathbf{m}_c(\mathbf{y}) + \mathbf{d}\} \cap \mathbb{R}_v(\Lambda_s') \tag{4.21}$$

Fundamental regions for the lattices in the partition chain $\Lambda_c/\Lambda_c'/\Lambda_s/\Lambda_s'$ can be formed from unions of fundamental regions of lattices further to the left in the chain. Let $\mathbb{R}(\Lambda_c)$ be a fundamental region for the finest lattice, Λ_c. Since $|\Lambda_c/\Lambda_c'| = 2^{n_c}$, a fundamental region for Λ_c' is

$$\mathbb{R}(\Lambda_c') = \bigcup_{\mathbf{y}} \{\mathbb{R}(\Lambda_c) + \mathbf{m}(\mathbf{y})\} \tag{4.22}$$

which contains 2^{n_c} points from Λ_c, one for each coset of Λ_c' in Λ_c.

EXAMPLE 4.1 (a) Fundamental Regions for $\mathbf{Z}^2$ and $2\mathbf{Z}^2$

Consider the partition chain $\Lambda_c/\Lambda_c'/\Lambda_s/\Lambda_s' = \mathbf{Z}^2/2\mathbf{Z}^2/8\mathbf{Z}^2/16\mathbf{Z}^2$. Let $\mathbf{y} = (y_1, y_2)$ with y_1 and y_2 taking values 0 or 1. In this example $n_c = 2$. Also let $\mathbf{d} = (0.5, 0.5)$. A Set of coset representatives for $\mathbf{Z}^2/2\mathbf{Z}^2$ is

$$[\mathbf{Z}^2/2\mathbf{Z}^2] = \{(0,0), (0,1), (1,0), (1,1)\} \tag{4.23}$$

Thus the mapping function from the coset label $\mathbf{y}$ to the coset representative is

$$\mathbf{m}_c(\mathbf{y}) = (y_1, y_2) \tag{4.24}$$

A fundamental region for $\Lambda_c = \mathbf{Z}^2$ is

$$\mathbb{R}(\Lambda_c) = \{(x,y) \,|\, 0 \le x < 1, 0 \le y < 1\} \tag{4.25}$$

According to (4.22) a fundamental region for $\Lambda_c' = 2\mathbf{Z}^2$ is

$$\begin{aligned} \mathbb{R}(2\mathbf{Z}^2) &= \{\mathbb{R}(\mathbf{Z}^2) + (0,0)\} \cup \{\mathbb{R}(\mathbf{Z}^2) + (0,1)\} \cup \{\mathbb{R}(\mathbf{Z}^2) + (1,0)\} \\ &\quad \cup \{\mathbb{R}(\mathbf{Z}^2) + (1,1)\} \\ &= \{(x,y) \,|\, 0 \le x < 2, 0 \le y < 2\} \end{aligned} \tag{4.26}$$

These regions are illustrated in Figure 4.3.

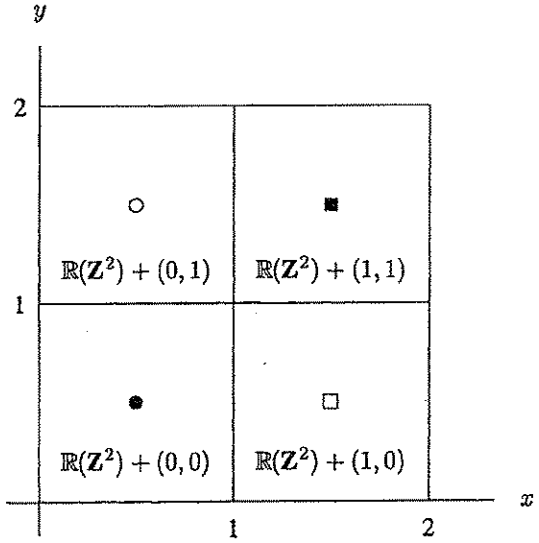


Figure 4.3. Fundamental Regions for $\mathbf{Z}^2$ and $2\mathbf{Z}^2$

For the next partition level, $|\Lambda_c'/\Lambda_s| = 2^{n_u}$ so a fundamental region for Λ_s is

$$\mathbb{R}(\Lambda_s) = \bigcup_{\mathbf{w}} \{\mathbb{R}(\Lambda_c') + \mathbf{m}_u(\mathbf{w})\} \tag{4.27}$$

Also, $|\Lambda_c/\Lambda_s| = 2^{n_c+n_u}$, so $\mathbb{R}(\Lambda_s)$ contains $2^{n_c+n_u}$ points from $\Lambda_c + \mathbf{d}$; one for each coset of Λ_s in Λ_c. The region $\mathbb{R}(\Lambda_s)$ contains 2^{n_u} points from each of the 2^{n_c} cosets of Λ_c' in Λ_c.

Consider the region $\mathbb{R}_v(\Lambda_s')$ defined on page 98 which is the Voronoi region for Λ_s' with redundant boundary points removed. Parts of $\mathbb{R}(\Lambda_s)$ may fall outside of $\mathbb{R}_v(\Lambda_s')$. These parts can be translated into $\mathbb{R}_v(\Lambda_s')$ by adding the appropriate element of Λ_s'. This region is described by the equation

$$\mathbb{R}^{(0)}(\Lambda_s) = \{\mathbb{R}(\Lambda_s) + \Lambda_s'\} \cap \mathbb{R}_v(\Lambda_s') \tag{4.28}$$

and is another fundamental region for Λ_s and so contains $2^{n_c+n_u}$ points from $\Lambda_c + \mathbf{d}$. We will call this region *shaping region* 0. The points of $\Lambda_c + \mathbf{d}$ in this region will be called the constellation $\mathbf{S}^{(0)}$ and can be described by the equation

$$\mathbf{S}^{(0)} = \mathbb{R}^{(0)}(\Lambda_s) \cap \{\Lambda_c + \mathbf{d}\} \tag{4.29}$$

EXAMPLE 4.1 (b) A Fundamental Region $\mathbb{R}(8\mathbf{Z}^2)$

Continuing with the same lattice partition chain, the second level partition $\Lambda_c'/\Lambda_s = 2\mathbf{Z}^2/8\mathbf{Z}^2$ has order $|2\mathbf{Z}^2/8\mathbf{Z}^2| = 2^4 = 16$ which requires that $n_u = 4$. Let $\mathbf{w} = (w_1, w_2, w_3, w_4)$ be a binary 4-tuple with the mapping function

$$\mathbf{m}_u(\mathbf{w}) = (4w_1 + 2w_2, 4w_3 + 2w_4) \in 2\mathbf{Z}^2 = \Lambda_c' \tag{4.30}$$

Using (4.27) to form a fundamental region for Λ_s, we find in this example that $\mathbb{R}(\Lambda_s) \subset R_v(\Lambda_s')$ so $\mathbb{R}^{(0)}(\Lambda_s) = \mathbb{R}(\Lambda_s)$ and $\mathbf{S}^{(0)}$ consists of the 64 points in the first quadrant of the form

$$\begin{aligned}(x, y) &= (4w_1 + 2w_2 + y_1 + 0.5, 4w_3 + 2w_4 + y_2 + 0.5) \\ &= \mathbf{m}_c(\mathbf{y}) + \mathbf{m}_u(\mathbf{w}) + (0.5, 0.5)\end{aligned} \tag{4.31}$$

In two's complement notation, the point coordinates are

$$x = [0w_1w_2y_1.1] \text{ and } y = [0w_3w_4y_2.1] \tag{4.32}$$

The fundamental region for Λ_s is shown in Figure 4.4.

■

The initial constellation points $\mathbf{r}$ are specified by $\mathbf{y}$, $\mathbf{w}$, and $\mathbf{t}$. Thus the initial constellation must contain $2^{n_c+n_s+r_s}$ points since $\mathbf{t}$ has r_s nonzero components. Let *shaping region* $\mathbf{t}$ be defined as

$$\begin{aligned}\mathbb{R}^{(\mathbf{t})}(\Lambda_s) &= \{\mathbb{R}^{(0)}(\Lambda_s) + \mathbf{m}_s(\mathbf{t}) + \Lambda_s'\} \cap \mathbb{R}_v(\Lambda_s') \\ &= \{\mathbb{R}(\Lambda_s) + \mathbf{m}_s(\mathbf{t}) + \Lambda_s'\} \cap \mathbb{R}_v(\Lambda_s')\end{aligned} \tag{4.33}$$

with the corresponding constellation

$$\mathbf{S}^{(\mathbf{t})} = \mathbb{R}^{(\mathbf{t})}(\Lambda_s) \cap \{\Lambda_c + \mathbf{d}\} \tag{4.34}$$

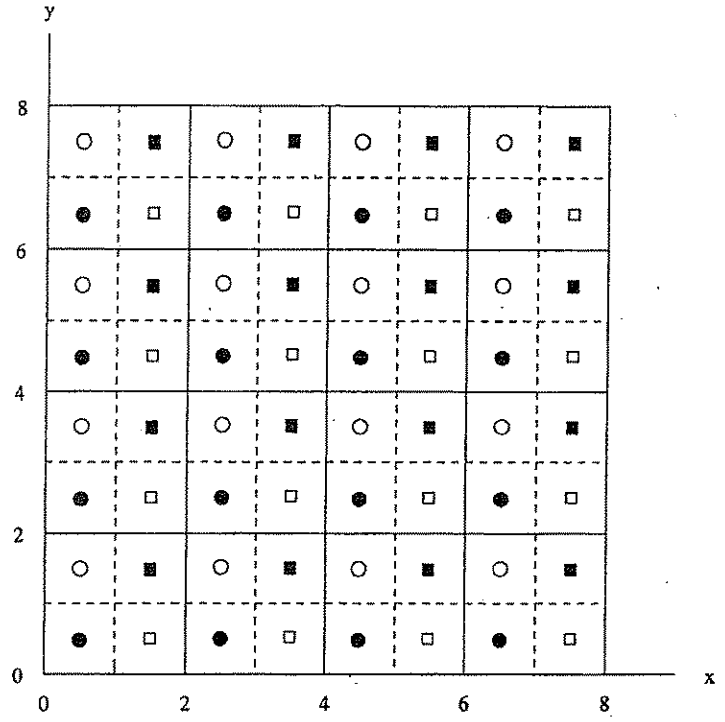


Figure 4.4. Fundamental Region $\mathbb{R}(8\mathbf{Z}^2)$ for $8\mathbf{Z}^2$

Each shaping region $\mathbb{R}^{(\mathbf{t})}(\Lambda_s)$ contains $2^{n_c+n_u}$ points of $\Lambda_c + \mathbf{d}$ so the union

$$\mathbf{A} = \bigcup_{\mathbf{t}} \mathbb{R}^{(\mathbf{t})}(\Lambda_s) \tag{4.35}$$

contains the $2^{n_c+n_u+r_s}$ points

$$\mathbf{S} = \bigcup_{\mathbf{t}} \mathbf{S}^{(\mathbf{t})} \tag{4.36}$$

These points can be used as the initial constellation from which $\mathbf{r}$ is selected and the initial mapping function can be selected to be

$$\mathbf{r} = \mathbf{m}(\mathbf{y}, \mathbf{w}.\mathbf{t}) = \{\Lambda'_s + \mathbf{m}_c(\mathbf{y}) + \mathbf{m}_u(\mathbf{w}) + \mathbf{m}_s(\mathbf{t}) + \mathbf{d}\} \cap \mathbf{A} \tag{4.37}$$

EXAMPLE 4.1 (c) Continuing the Same Example for Λ_s/Λ'_s

At the third lattice partitioning level, $|\Lambda_s/\Lambda'_s| = |8\mathbf{Z}^2/16\mathbf{Z}^2| = 4$ so $n_s = 2$. We will use the inverse syndrome former $(\mathbf{H}_s^t)^{-1}$ for the Ungerboeck 4-state code presented in Example 3.9 and shown in Figure 3.13, so $\mathbf{t} = (t_1, 0)$. Let the mapping function for the coset representatives $[8\mathbf{Z}^2/16\mathbf{Z}^2]$ be

$$\mathbf{m}_s(\mathbf{z}) = -8(z_1, z_2) \in 8\mathbf{Z}^2 = \Lambda_s \tag{4.38}$$

Then

$$\mathbb{R}^{(1,0)}(\Lambda_s) = \mathbb{R}(\Lambda_s) + (-8, 0) = \{(x, y) \mid -8 \le x < 0, 0 \le y < 8\} \tag{4.39}$$

and

$$\mathbf{r} = \mathbf{m}(\mathbf{y}, \mathbf{w}, \mathbf{t}) = (-8t_1 + 4w_1 + 2w_2 + y_1 + 0.5, 4w_3 + 2w_4 + y_2 + 0.5) \tag{4.40}$$

The constellation at the output of the memoryless map, which is also called the initial constellation, contains $2^7 = 128$ points. The 2's complement representations for the point coordinates are

$$x = (t_1\, w_1\, w_2\, y_1 \,.\, 1) \text{ and } y = (0\, w_3\, w_4\, y_2 \,.\, 1) \tag{4.41}$$

The initial constellation is shown in Figure 4.5.

■

The decoder for $\mathbb{C}_s$ changes the initial constellation into a final constellation by quantizing the initial sequence $\mathbf{r}(D)$ to the sequence $\mathbf{c}(D)$ in the shaping code that is closest in squared Euclidean distance and transmitting the quantization error $\mathbf{a}(D)$. At each symbol time the shaping decoder outputs a binary n_s-tuple

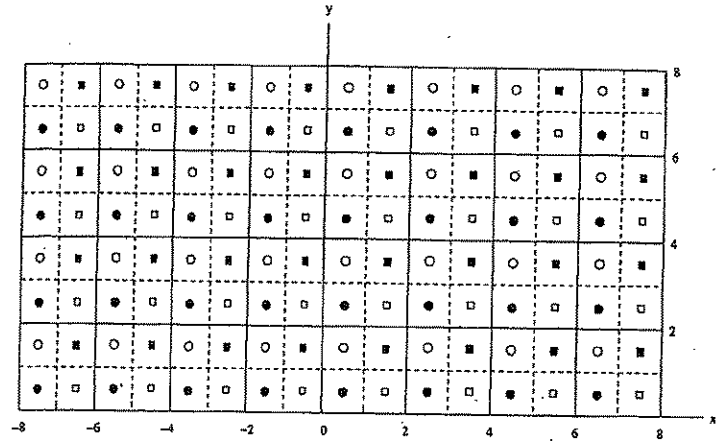


Figure 4.5. The Initial Constellation with 128 Points

$\mathbf{v}$, and a point in $\mathbb{R}_v(\Lambda'_s)$ is selected from the coset of Λ'_s with the representative $\mathbf{m}_s(\mathbf{v})$. This point has the form

$$\mathbf{c} = \lambda'_s + \mathbf{m}_s(\mathbf{v}) \text{ with } \lambda'_s \in \Lambda'_s \tag{4.42}$$

The final transmitted point is

$$\begin{aligned} \mathbf{a} &= \mathbf{r} - \mathbf{c} = \mathbf{m}(\mathbf{y}, \mathbf{w}, \mathbf{t}) - \lambda'_s - \mathbf{m}_s(\mathbf{t}) \\ &= \tilde{\lambda}'_s + \mathbf{m}_s(\mathbf{t}) + \mathbf{m}_u(\mathbf{w}) + \mathbf{m}_c(\mathbf{y}) + \mathbf{d} - \lambda'_s - \mathbf{m}_s(\mathbf{v}) \\ &= \tilde{\lambda}'_s + \mathbf{m}_s(\mathbf{t} \oplus \mathbf{v}) + \mathbf{m}_u(\mathbf{w}) + \mathbf{m}_c(\mathbf{y}) + \mathbf{d} \end{aligned} \tag{4.43}$$

Therefore, $\mathbf{a}$ must be in the shaping region $\mathbb{R}^{(\mathbf{z})}(\Lambda_s)$ where $\mathbf{z} = \mathbf{t} \oplus \mathbf{v}$. The n_s-tuple $\mathbf{t}$ has only r_s nonzero components. However, the components of $\mathbf{v}$ can have all possible values, so $\mathbf{z}$ can also have all 2^{n_s} values. Thus the final constellation falls in one of the initial 2^{r_s} shaping regions when $\mathbf{v}$ is zero in the $k_s = n_s - r_s$ fixed zero positions of $\mathbf{t}$, but falls in additional $2^{n_s} - 2^{r_s}$ shaping regions when $\mathbf{v}$ alters the fixed 0 positions in $\mathbf{t}$. Since $|\Lambda_s/\Lambda'_s| = 2^{n_s}$,

$$\mathbb{R}_v(\Lambda'_s) = \bigcup_{\mathbf{z}} \mathbb{R}^{(\mathbf{z})}(\Lambda_s) \tag{4.44}$$

and contains the required $2^{n_c+n_u+n_s}$ points from $\Lambda_c + \mathbf{d}$.

EXAMPLE 4.2 (d) Continuing the Example to Find the Final Constellation

For this same example we defined the function for mapping coset labels of Λ'_s

into coset representatives to be $\mathbf{m}_s(\mathbf{v}) = -8(v_1, v_2)$ so

$$\begin{aligned}\mathbf{a} &= \mathbf{r} - \mathbf{c} = (-8t_1 + 4w_1 + 2w_2 + y_1 + 0.5, 4w_3 + 2w_4 + y_2 + 0.5) \\ &\quad + 8(v_1, v_2) + 16(i_1, i_2)(16i_1 - 8t_1 + 8v_1 + 4w_1 + 2w_2 + y_1 + 0.5, \\ &\quad 16i_2 + 8v_2 + 4w_3 + 2w_4 + y_2 + 0.5) \\ &\quad \in \mathbb{R}_v(16\mathbf{Z}^2) \end{aligned} \tag{4.45}$$

The binary 2-tuple (i_1, i_2) is chosen to put $\mathbf{a}$ in the Voronoi region of $16\mathbf{Z}^2$. For $\mathbf{a}$ to be in $\mathbb{R}_v(16\mathbf{Z}^2)$ we need to select i_1 so that

$$-8 \le 16i_1 - 8(t_1 - v_1) < 8 \tag{4.46}$$

The required values for i_1 are shown in Table 4.1.

Table 4.1. Selecting i_1 so that $\mathbf{a} \in \mathbb{R}_v(16\mathbf{Z}^2)$

t_1	v_1	$t_1 - v_1$	i_1	$16i_1 - 8(t_1 - v_1)$	$-8(t_1 \oplus v_1)$
0	0	0	0	0	0
1	1	0	0	0	0
1	0	1	0	-8	-8
0	1	-1	-1	-8	-8

We need to select i_2 so that

$$-8 \le 16i_2 + 8v_2 < 8 \tag{4.47}$$

The required values of i_2 are shown in Table 4.2. (Remember that t_2 is always 0.)

Table 4.2. Selecting i_2 so that $\mathbf{a} \in \mathbb{R}_v(16\mathbf{Z}^2)$

t_2	v_2	i_2	$16i_2 + 8v_2$	$-8(t_2 \oplus v_2)$
0	0	0	0	0
0	1	-1	-8	-8

Thus the final point can be expressed as

$$\begin{aligned}\mathbf{a} &= [-8(t_1 \oplus v_1) + 4w_1 + 2w_2 + y_1 + 0.5, -8(t_2 \oplus v_2) \\ &\quad + 4w_3 + 2w_4 + y_2 + 0.5] \end{aligned} \tag{4.48}$$

So in 2's complement notation the coordinates of $\mathbf{a}$ are

$$x = [(t_1 \oplus v_1)\, w_1\, w_2\, y_1\, .\, 1] \text{ and } y = [v_2\, w_3\, w_4\, y_2\, .\, 1] \tag{4.49}$$

Notice that the modifications caused by the decoder for the shaping code only affect the sign bits. Therefore, Forney calls this method *sign bit shaping*. The final constellation is illustrated in Figure 4.6. The four shaping regions $\mathbb{R}^{(z)}(8\mathbf{Z}^2)$ are the 8×8 squares in each quadrant. The arrows show how any point selected by $\mathbf{y}$ and $\mathbf{w}$ in the first quadrant can be modified by the shaping operation.

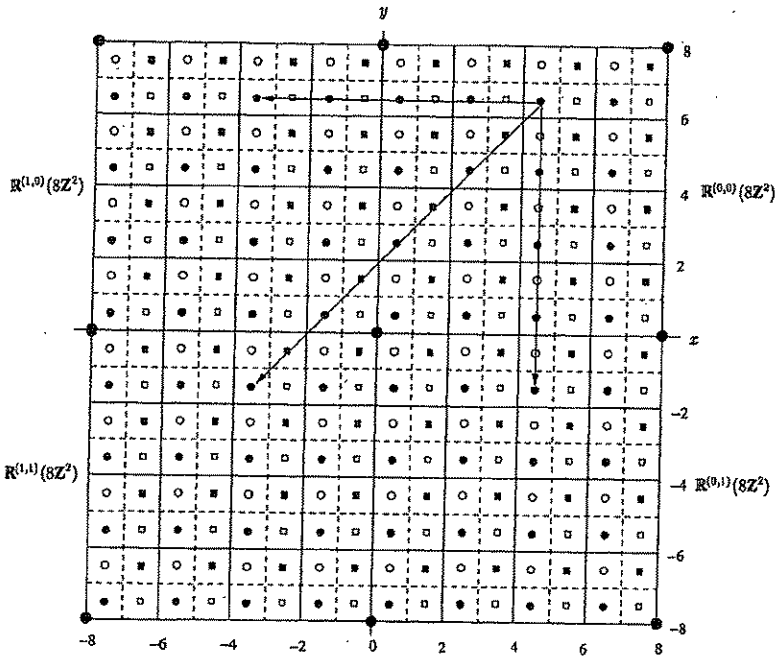


Figure 4.6. The Final 256-Point Constellation

The final constellation has $M_1 = 2^{n_c+n_u+n_s}$ points. An unshaped constellation with $n_c+n_u+r_s$ bits per N-dimensional symbol requires $M_2 = 2^{n_c+n_u+r_s}$ points, so the shaping constellation expansion ratio is

$$\text{CER}_s = M_1/M_2 = 2^{k_s} \text{ per } N \text{ dimensions} \tag{4.50}$$

The points in the final constellation do not occur with equal likelihood. Points closer to the origin are significantly more likely than ones farther away. This is a

result of the shaping operation which minimizes the energy in the final sequence $\mathbf{a} = \mathbf{r} - \mathbf{c}$. Whenever possible, the shaping decoder selects a point for $\mathbf{c}$ that causes $\mathbf{a}$ to be as small as possible. Forney [23] has experimentally measured the one-dimensional probability densities for the transmitted coordinates and found them to be nearly Gaussian. He reports shaping gains of about 0.8 dB using the simple 4-state Ungerboeck code of the above example with a shaping decoder delay of 20 2D symbols. The shaping gain increases to this value asymptotically with shaping decoder delay.

EXAMPLE 4.3 Another Purely Lattice Theoretical Partitioning

Consider the same partition chain

$$\Lambda_c/\Lambda'_s/\Lambda_s/\Lambda'_s = \mathbf{Z}^2/2\mathbf{Z}^2/8\mathbf{Z}^2/16\mathbf{Z}^2$$

and the same label to coset representative mappings as in the previous example. Another fundamental region for Λ_c shown in Figure 4.7 is

$$\mathbb{R}(\Lambda_c) = \mathbb{R}(\mathbf{Z}^2) = \{(x,y) \mid -4 \le x < -3, -4 \le y < -3\} \tag{4.51}$$

Then

$$\mathbb{R}(\Lambda'_c) = \mathbb{R}(2\mathbf{Z}^2) = \{-4 \le x < -2, -4 \le y < -2\} \tag{4.52}$$

and

$$\mathbb{R}^{(0,0)}(\Lambda_s) = \mathbb{R}(8\mathbf{Z}^2) = \{(x,y) \mid -4 \le x < 4, -4 \le y < 4\} \tag{4.53}$$

The shaping regions are

$$\begin{aligned} \mathbb{R}^{(\mathbf{z})}(\Lambda_s) &= \{\Lambda'_s + \mathbb{R}^{(0,0)}(\Lambda_s) + \mathbf{m}_s(\mathbf{z})\} \cap \mathbb{R}_v(\Lambda'_s) \\ &= \{16\mathbf{Z}^2 + \mathbb{R}^{(0,0)}(8\mathbf{Z}^2) - 8(z_1, z_2)\} \\ &\quad \cap \{(x,y) \mid -8 \le x < 8, -8 \le y < 8\} \end{aligned} \tag{4.54}$$

■

4.2 Trellis Shaping on Regions

Around the time Forney presented his work on trellis shaping at the Information Theory Workshop at Cornell University in June 1989, he became aware of the work of Calderbank and Ozarow [8] at Bell Laboratories in Murray Hill, N.J. He saw how trellis shaping based on lattice partitions could be generalized to include Calderbank and Ozarow's idea of shaping on regions which resulted in a more flexible and easier to understand scheme. In subsequent versions of his paper [23], Forney switched to first introducing his ideas through trellis shaping on regions with an appendix describing shaping based on lattice partitions. In this section, the essential properties of trellis shaping based on lattice partitions are presented first. Then the generalization to trellis shaping on regions consistent with these properties is discussed.

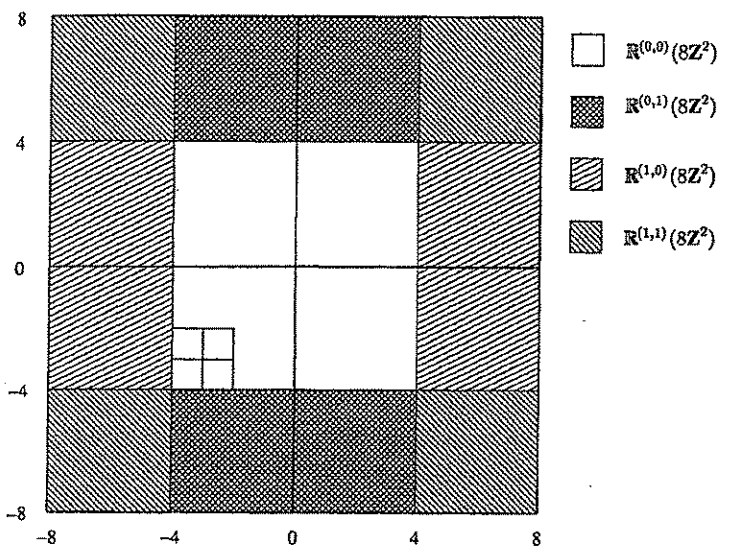


Figure 4.7. The Shaping Regions for Another Lattice Theoretical Partitioning

4.2.1 Essential Properties of Trellis Shaping Based on Lattice Partitions

The essential properties of trellis shaping based on lattice partitions are

1 A $2^{n_c+n_u+n_s}$ point constellation constrained to a region $\mathbb{R}$ of N-space.

2 A partition of $\mathbb{R}$ into 2^{n_s} shaping regions $\mathbb{R}^{(\mathbf{z})}$, each containing $2^{n_c+n_u}$ points which are 2^{n_u} points from each of the 2^{n_c} cosets of the channel coding sublattice Λ'_c.

3 The transmitted sequence is selected as the minimum energy sequence in the equivalence class

$$\mathbf{S}(\mathbf{y}, \mathbf{w}, \mathbf{t}) = \{\mathbf{m}(\mathbf{y}, \mathbf{w}, \mathbf{z}) \mid \mathbf{z} = \mathbf{t} \oplus \mathbf{v}, \mathbf{v} \in C_s\} \tag{4.55}$$

4 The constellation mapping function $\mathbf{m}(\mathbf{y}, \mathbf{w}, \mathbf{z})$ has the properties:

(a) $\mathbf{m}(\mathbf{y}, \mathbf{w}, \mathbf{z})$ for fixed $\mathbf{y}$ is always in the channel code coset $\Lambda'_c + \mathbf{m}_c(\mathbf{y})$ for all $\mathbf{z}$ and $\mathbf{w}$.

(b) $\mathbf{m}(\mathbf{y}, \mathbf{w}, \mathbf{z})$ for fixed $\mathbf{z}$ is always in the shaping region $\mathbb{R}^{(\mathbf{z})}$ as $\mathbf{y}$ and $\mathbf{w}$ vary.

Notice that property 2 does not require the shaping regions to be translations of a fundamental region for Λ_s as described for shaping based on lattice partitions. The shaping regions can be arbitrary as long as the requirements of property 2 are satisfied. Also, the $\mathbf{w}$ bits do not have to be assigned to points by a lattice theoretical mapping. They can be assigned in any convenient way as long as properties 2 and 4 are satisfied.

EXAMPLE 4.4 Sign Bit Shaping

Let $\mathbf{y}$ have dimension $n_c = 2$, $\mathbf{w}$ dimension $n_u = 4$, and $\mathbf{z}$ dimension $n_s = 2$ as in Example 4.1. The 8×8 square regions in each of the four quadrants shown in Figure 4.6 can be used as the shaping regions. The channel encoder output 2-tuple, $\mathbf{y}$, selects one of the four cosets, ○, ●, □, and ■. Within the shaping region in the first quadrant, $\mathbb{R}^{(0,0)}$, the 4-tuples $\mathbf{w}$ can be assigned to the 16 points in each of the four cosets in any convenient manner. In the case of sign bit shaping, the first quadrant points are mapped into the three other shaping regions by changing the sign bits of their x and y coordinates to $\mathbf{z} = \mathbf{t} \oplus \mathbf{v}$ where $\mathbf{v}$ is the binary code sequence in C_s chosen by the decoder for $\mathbb{C}_s$ to minimize the energy in the transmitted sequence $\mathbf{a}$, and $\mathbf{t}$ is the output of the inverse syndrome former for C_s. ■

EXAMPLE 4.5 Shaping Regions for 90° Rotational Invariance

A set of four shaping regions for the case where $\mathbf{z}$ is a 2-tuple is shown in Figure 4.8. Notice that a 90° rotation maps each shaping region onto itself. Therefore, the shaping bits, $\mathbf{z}$, are transparent to 90° rotations. One quarter of each shaping region is in the first quadrant. Each shaping region is the union of the quarter in the first quadrant, and its 90°, 180°, and 270° rotations. In addition, each quarter can be further partitioned into four 2×2 squares shown by the dotted lines in Figure 4.6. Consider the partition of $\mathbf{Z}^2 + (0.5, 0.5)$ into four subsets as shown in Figure 4.6. Then, each 2×2 square contains four points, one from each translated coset of $2\mathbf{Z}^2$. Each distinct 4-tuple, $\mathbf{w}$, can be assigned to one of the sixteen 2×2 squares in the first quadrant. Then each 2×2 square can be rotated about the origin by multiples of 90° and the same $\mathbf{w}$ assigned to each rotation. Thus, each shaping region will contain 64 points satisfying properties 2 and 4, and $\mathbf{w}$ will be transparent to 90° rotations. An example of this assignment is shown in Figure 4.9 where each 2×2 square is labelled by the 4-tuple $\mathbf{w} = (w_1, w_2, w_3, w_4)$. The $\mathbf{x}$ bits decoded from the $\mathbf{y}$ bits must be made rotationally invariant by the properties of the channel convolutional code. The Wei 16-state 4D code has the required rotational invariance.

■

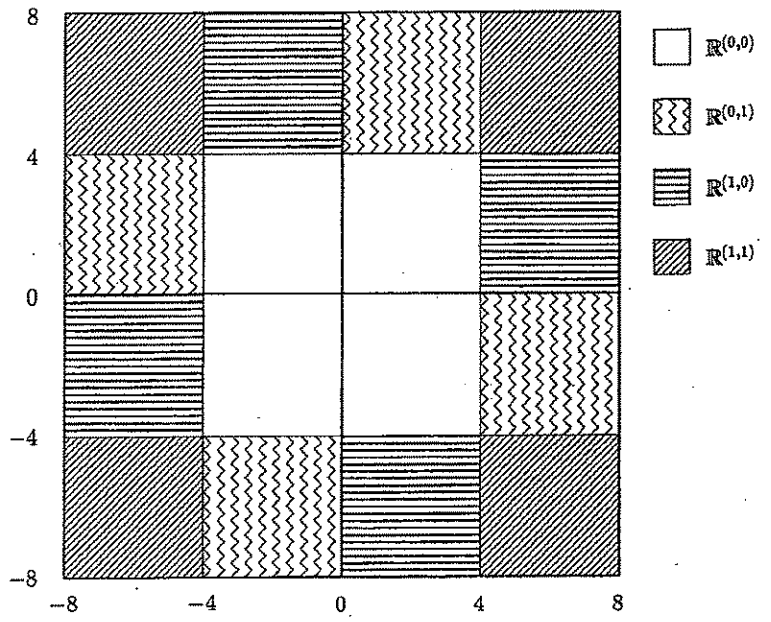


Figure 4.8. Shaping Regions for 90° Rotational Invariance

4.2.2 The Trellis Shaping Encoder for Shaping on Regions

The block diagram for the encoder for shaping on regions is shown in Figure 4.10. Its components are described in the following subsections. This encoder has essentially the same complexity as the encoder for shaping based on lattice partitions.

4.2.2.1 The Channel Trellis Code and Inverse Syndrome Former

The channel trellis code and inverse syndrome former blocks are exactly the same as the ones in Figure 4.1 which shows the encoder for shaping based on lattice partitions.

4.2.2.2 The Decoder for $\mathbb{C}_s$

Remember that the transmitted sequence is the minimum energy sequence in the equivalence class

$$\mathbf{S}(\mathbf{y}, \mathbf{w}, \mathbf{t}) = \{\mathbf{m}(\mathbf{y}, \mathbf{w}, \mathbf{z}) \,|\, \mathbf{z} = \mathbf{t} \oplus \mathbf{v}, \mathbf{v} \in \mathbb{C}_s\} \tag{4.56}$$

The decoder for $\mathbb{C}_s$ finds the shaping code sequence $\mathbf{v}$ that gives the minimum energy transmitted sequence. The equivalence class of possible transmitted

1111	1110	1101	1100	0011	0111	1011	1111
1011	1010	1001	1000	0010	0110	1010	1110
0111	0110	0101	0100	0001	0101	1001	1101
0011	0010	0001	0000	0000	0100	1000	1100
1100	1000	0100	0000	0000	0001	0010	0011
1101	1001	0101	0001	0100	0101	0110	0111
1110	1010	0110	0010	1000	1001	1010	1011
1111	1011	0111	0011	1100	1101	1110	1111

8 4 0 -4 -8 (rows)
-8 -4 0 4 8 (columns)

Figure 4.9. Assignments of $\mathbf{w} = (w_1, w_2, w_3, w_4)$ for 90° Rotational Invariance

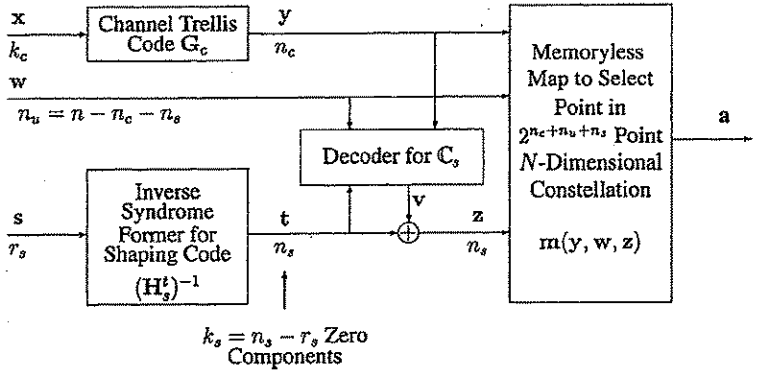


Figure 4.10. Encoder for Trellis Shaping on Regions

sequences can be represented by the same trellis diagram as for the binary shaping code C_s except that each branch is labelled by the constellation point $\mathbf{m}(\mathbf{y}, \mathbf{w}, \mathbf{t} \oplus \mathbf{v})$ instead of by the binary n_s-tuple $\mathbf{v}$. Thus the branch labels change with time as $\mathbf{y}$, $\mathbf{w}$, and $\mathbf{t}$, change according to the input bits. The Viterbi algorithm can be used to find the optimum $\mathbf{v}$. The decoder uses the squared magnitude of the current value of $\mathbf{m}(\mathbf{y}, \mathbf{w}, \mathbf{z})$ as the branch metric and the sum of the branch metrics along a path as the cumulative path metric.

4.2.2.3 Warning About the Viterbi Decoder Implementation

Error free data recovery in the receiver depends on the fact that $\mathbf{v}$ is a sequence in the trellis shaping code C_s so that

$$\mathbf{z}\mathbf{H}_s^t = (\mathbf{t} \oplus \mathbf{v})\mathbf{H}_s^t = \mathbf{t}\mathbf{H}_s^t \oplus \mathbf{v}\mathbf{H}_s^t = \mathbf{s} \tag{4.57}$$

Many Viterbi decoder implementations do not force the decoded sequence to be a true trellis sequence. Typically, a finite window of past decisions is stored in the decoder's trellis memory and the output bits are determined by starting at the most likely current trellis state and tracing a path back through the trellis memory to the end of the window. The bits at the end of the window are used as the decoder's output. This does not insure that the output sequence is a connected trellis path. A break in the trellis path would normally occur only under very noisy channel conditions. In trellis shaping, the signal observed by the shaping decoder comes from a finely space lattice Λ_c while the shaping code sequences come from a coarser sublattice Λ_s, so this is like decoding a signal at the output of a noisy channel.

A strategy for keeping the trellis paths connected will now be presented. Suppose a shaping code with M states is used and that the surviving path trellis memory has a depth of d nodes. The record for each node contains M fields, one for each state at that trellis depth. Assume that the field for each state in a record contains a pointer to the state at the previous node that is on the surviving path to the state corresponding to this field. In addition to the trellis memory, let another record consisting of the last state on the path output by the shaping decoder be saved.

As usual, the first step in a decoding iteration is to compute the survivors to each state at the current trellis depth and their path metrics and save pointers in the trellis memory record and path metrics in another array.

Now the surviving paths must be tested to make sure they are connected to the previously output path, so the next step is to trace each of the current M states back to the end of the trellis memory. If the pointers determined by the trace back operation for each of the M paths are all equal to the value stored in the "last state record," then all M surviving paths are connected to the previously output path. Select the current state with the best (smallest) path metric and use the $\mathbf{v}$ determined by tracing back this state to the end of the trellis memory as

the new decoder output and store the state reached at the end of the trace back operation in the last state record.

If some, but not all, of the surviving paths connect to the previously output path, choose the connected path with the smallest path metric and use its traced back $\mathbf{v}$ as the output and store the corresponding state in the last state record. For each of the unconnected paths, set the path metric to a very large number. This will cause these paths to be rejected in future decoding iterations.

A connected path will always be chosen by this simple technique. At a particular iteration, when paths converging on a given state are compared, any unconnected path will be rejected relative to a connected path because of its large path metric. If only unconnected paths converge on a state, an unconnected path will be chosen and the updated path metric will still be very large. When only connected paths converge on a state, a connected path will be selected. These are the only three possibilities. Thus, if a connected path is one of the survivors in the previous iteration, one of the survivors for the current iteration must also be a connected path.

4.2.3 The Receiver for Shaping on Regions

The receiver for the shaping on regions method is exactly the same as the one shown in Figure 4.2 for shaping based on lattice partitions.

4.2.4 Peak-to-Average Ratio Considerations

Trellis shaped constellation usually have a higher peak-to-average power ratio (PAR) than unshaped ones. Forney [23] has investigated methods for reducing the PAR. One approach was to choose the shaping regions as shells bounded by concentric N-spheres. (See Figure 10 of [23].) Each shell contains $2^{n_c+n_u}$ points so each must have the same volume. Calderbank and Ozarow [8] also studied these kind of regions. By restricting the shaping decoder from choosing points in the outer shell, the PAR was reduced by a factor of 3/4 but the measured shaping gain decreased by only 0.03 dB from 1.03 to 1.00 dB.

4.2.5 CER_s and PAR_2 Constraints with the 4-State Ungerboeck Shaping Code

In this section, methods for limiting the shaping constellation expansion ratio (CER_s) and peak-to-average ratio (PAR) in trellis shaping systems are discussed using Forney's form of the Ungerboeck 4-state code for the shaping code as an example. A block diagram of the encoder for this code is shown in Figure 3.11. Let $\mathbf{v} = (c_1, c_2)$ be the encoder output which is the pair of bits assigned to a trellis branch and also specifies the coset representative for the shaping code lattice partition Λ_s/Λ'_s. Two branches leave each state in the trellis of the Ungerboeck 4-state code. The branch labels for each pair must be:

(1) $\{(00),(11)\}$, or (2) $\{(0,1),(1,0)\}$. That is, the labels on the two branches leaving a node must be the logical complements of each other. Let the original $2^{n_c+n_u}$ point constellation be confined to a square $\mathbb{R}^{(0,0)}$ of side a in the first quadrant and let the shaping sublattice be $\Lambda'_s = a\mathbf{Z}^2$. Then, for an initial point $(x,y) \in \mathbb{R}^{(0,0)}$, the shaped point can become

(1): (x,y) or $(x,y) - a(1,1)$

or

(2): $(x,y) - a(0,1)$ or $(x,y) - a(1,0)$

For case (1), the shaped points lie in the original square $\mathbb{R}^{(0,0)}$ or in $\mathbb{R}^{(1,1)}$ which is the translation of $\mathbb{R}^{(0,0)}$ to the third quadrant, that is, $\mathbb{R}^{(1,1)} = \mathbb{R}^{(0,0)} - a(1,1)$. If the closet point to the origin is selected from each pair $\{(x,y),(x,y) - a(1,1)\}$ the $2^{n_c+n_u}$ points in the shaded region of Figure 4.11 are left. The shaded region of one square is the translate of the unshaded region of the other.

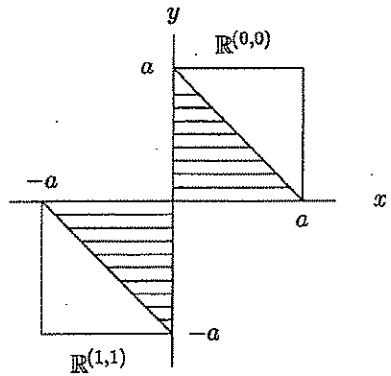


Figure 4.11. Regions for Case (1): $\mathbf{v} = (0,0)$ or $(1,1)$

For case (2), $\mathbb{R}^{(0,0)}$ is translated to $\mathbb{R}^{(1,0)} = \mathbb{R}^{(0,0)} - a(1,0)$ or $\mathbb{R}^{(0,1)} = \mathbb{R}^{(0,0)} - a(0,1)$. If the closest point to the origin in each pair $\{(x,y) - a(0,1),(x,y) - a(1,0)\}$ is selected, the $2^{n_c+n_u}$ points in the shaded region of Figure 4.12 are left.

The union of the two shaded regions is shown in Figure 4.13. It contains $2^{n_c+n_u+1}$ constellation points.

4.2.5.1 CER_s Constraints

With no constraints, the entire $(2a) \times (2a)$ square in Figure 4.13 containing $2^{n_c+n_u+2}$ constellation points is used and $\mathrm{CER}_s = 2$. If points are restricted to lie in the inscribed shaded square, $\mathrm{CER}_s = 1$. The shaping decoder will not

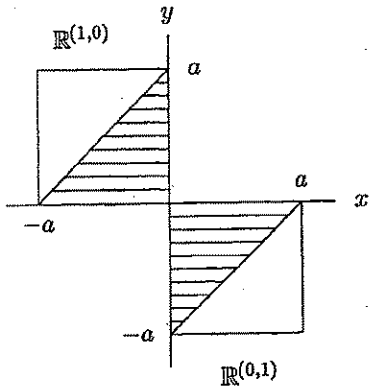


Figure 4.12. Regions for Case (2): $\mathbf{v} = (0,1)$ or $(1,0)$

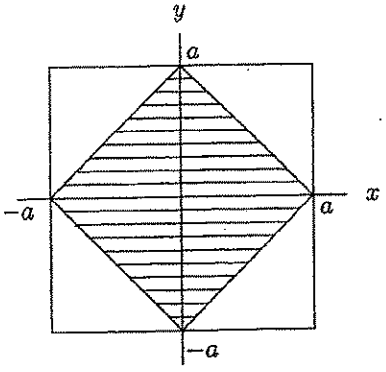


Figure 4.13. Union of Regions for Cases (1) and (2)

fail in this case because one point from each pair of paths leaving each trellis state is present so the trellis path can be extended by the one branch leaving the state. If a constellation which is a subset of the inscribed square is used, the decoder will have dead ends since for some $(\mathbf{y}, \mathbf{w})$ and state, no branches will be able to leave the state. Thus, by selecting a constellation that is the union of the inscribed inner square and some points in the outer square, CER_s can be varied from 1 to 2.

For a circle of radius a centered at the origin as shown in Figure 4.14, using the continuous approximation gives

$$\text{CER}_s = \frac{\pi a^2}{(\sqrt{2}a)^2} = \frac{\pi}{2} = 1.57 \tag{4.58}$$

Simulations [18] have shown that using the circle constraint reduces the shaping gain by only 0.05 dB.

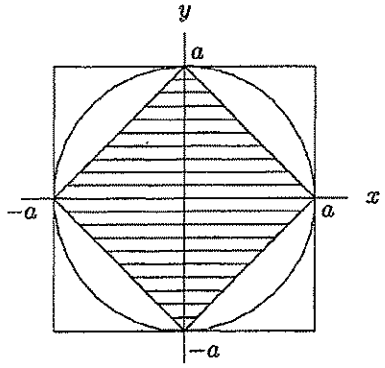


Figure 4.14. Constellation Expansion by a Circle

4.2.5.2 Peak Constraints

Peak power can be constrained by selecting a constellation which is the intersection of a circle for radius r centered at the origin and the outer square. When $r = \sqrt{2}a$ the circle inscribes the outer square, so for $r > \sqrt{2}a$ no constraint takes place and the entire square is used.

For $r = a$ the circle inscribes the inner square. This is a lower bound for r since for smaller values, points in the inner square are eliminated and the shaping decoder will run into dead ends.

For $a \leq r \leq \sqrt{2}a$ all points in the inner square and some additional points in the outer square which satisfy the peak power constraint are used.

EXHIBIT E
Part 3

Chapter 5

NONLINEAR PRECODING METHODS TO REMOVE INTERSYMBOL INTERFERENCE

Two nonlinear precoding methods that can be used in transmitters to remove intersymbol interference are presented in this chapter. The first method was disclosed independently by Tomlinson [58] in 1971 and Harashima [32] in 1972. The second method was discovered by Laroia, Tretter, and Farvardin (LTF) [47, 48] in 1992. Both methods transmit a signal that is the sum of the input constellation point sequence and a dither sequence that depends on the channel impulse response and precoder input. The advantage of the LTF precoder is that the dither signal is much smaller than for the Tomlinson/Harashima precoder so that the shaping gain of the input constellation is essentially preserved. The Tomlinson/Harashima precoder destroys shaping gain. Shortly after the LTF precoder was presented to the ITU-T V.34 study committee, additional refinements were made [43, 30] that further reduced the dither signal power by arranging the constellation point selector, precoder, and channel trellis encoder in a feedback loop and the technique was incorporated in Recommendation V.34. This refinement is the subject of Chapter 10.

A possible alternative to nonlinear precoding is to use a linear filter at the transmitter to perform what is often called compromise equalization or pre-emphasis. The frequency response of the linear filter is chosen to approximate the reciprocal of the channel response. Perfect linear equalization of some channels can require an unstable filter, and for channels that roll off steeply at the band edges, the pre-emphasis filter can cause the transmitted signal to have a large peak-to-average ratio. The nonlinear precoding methods are always stable and do not exaggerate the transmitted peak-to-average power ratio.

Actually, precoding was not proposed for channel equalization by the V.34 study committee. It was included to compensate for a noise whitening filter before the Viterbi decoder in the receiver. Noise whitening is discussed at the end of this chapter. Typically, an FIR adaptive equalizer is employed at the receiver

for channel equalization. The adaptive equalizer causes noise enhancement, or equivalently, introduces correlation in the noise, which is then removed by the noise whitening filter.

A simple block diagram of a system with precoding is shown in Figure 5.1. A sequence of ideal 2D constellation points, $a(n)$, is applied to the precoder resulting in the sequence $b(n)$. The precoder output is transmitted through a channel with the transfer function $H(z)$ whose output is the received sequence $r(n)$. The block labeled "Inverse Mapper" converts $r(n)$ back into the transmitted sequence $a(n)$ in a simple manner.

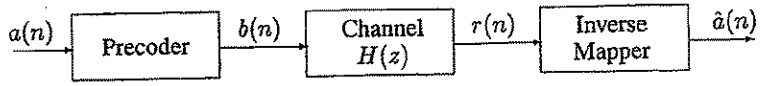


Figure 5.1. General Block Diagram of a System with Precoding

If the channel inverse $1/H(z)$ is stable and causal, it can be used as the Precoder block to exactly equalize the channel and the Inverse Mapper is not required. However, this is rarely possible for real-world channels. Cascading a linear, time-invariant filter with the transmitter to compensate for the channel response is sometimes called pre-emphasis.

5.1 Tomlinson/Harashima Precoding

One method for nonlinear precoding was discovered independently and at about the same time by Tomlinson [58] and Harashima and Miyakawa [32]. A block diagram of this precoder is shown in Figure 5.2.

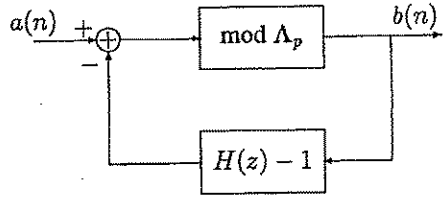


Figure 5.2. Basic Tomlinson/Harashima Precoder Block Diagram

The original unprecoded transmitted signal points, $a(n)$, are required to be contained in a fundamental region of a *precoding lattice* Λ_p. For example, in the articles by Tomlinson and Harashima only one-dimensional transmission systems are considered and an M-point constellation with M even can be chosen to be the M points of the translated 1D lattice $\mathbf{Z} + 0.5$ contained in the region $\mathbb{R}(\Lambda_p) = (-M/2, M/2]$ which is a fundamental region of the 1D precoding lattice $\Lambda_p = M\mathbf{Z}$. These are the half-integer points $n + 0.5$ for

$n = -M/2, \ldots, M/2 - 1$. A 2D example used in Eyubuğlu and Forney [18] is an M^2 point constellation consisting of the points from $\mathbf{Z}^2 + (0.5, 0.5)$ contained in the region $(-M/2, M/2] \times (-M/2, M/2]$ which is a fundamental region for the 2D lattice $M\mathbf{Z}^2$.

When the precoder is used in the simple cascaded structure shown in Figure 5.1 and trellis coding is not used to select the precoder input sequence $a(n)$, it is only necessary that the original constellation, that is, the set of points from which $a(n)$ is selected, be contained in a fundamental region $\mathbb{R}(\Lambda_p)$ of the precoding lattice Λ_p. When a trellis code based on a lattice partition Λ_c/Λ_c' is used to select the points $a(n)$, the precoding lattice Λ_p must be a sublattice of the coding sublattice Λ_c'. If the original constellation contains M points, the order of the partition of Λ_c with respect to Λ_p must be $|\Lambda_c/\Lambda_p| = M$.

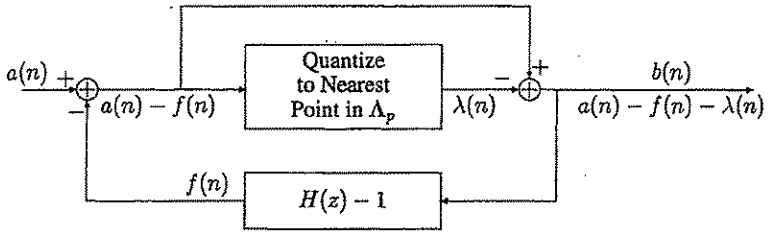


Figure 5.3. A More Detailed Precoder Block Diagram

The mod Λ_p block performs the operation shown in the more detailed precoder block diagram of Figure 5.3. The mod box quantizes its input $a(n) - f(n)$ to the nearest point $\lambda(n)$ in the precoding lattice Λ_p. It then subtracts the quantized value from the actual input to form the output signal $b(n)$ which is the quantization error. Thus $b(n)$ is limited in size because it must lie in the Voronoi region of Λ_p which is centered about the origin. To eliminate ambiguities that occur when an input point is equidistant from two or more precoding lattice points, the quantization error should be restricted to fall within a fundamental region of Λ_p that is contained in the Voronoi region of Λ_p.

For example, consider the M-point 1D constellation of half-integer points presented above. If the input to the mod Λ_p box lies exactly between the two precoding lattice points iM and $(i+1)M$, we can use the rule of quantizing it to the algebraically smaller lattice point. This point would be iM for $i \geq 0$. Then the quantization error $b(n)$ is confined to the region $(-M/2, M/2]$ which is a fundamental region of $M\mathbf{Z}$ and is contained in the Voronoi region $[-M/2, M/2]$.

For the M^2 point 2D constellation discussed above, the x and y components of the input can be independently quantized to the nearest point in $M\mathbf{Z}$ using the 1D rule of the previous paragraph. Then the 2D quantization error is confined

to the region

$$(-M/2, M/2] \times (-M/2, M/2]$$

which is a fundamental region of $M\mathbf{Z}^2$ contained in its Voronoi region

$$[-M/2, M/2] \times [-M/2, M/2]$$

The precoder output $b(n)$ is fed back to the input through the filter $H(z) - 1$. Thus, somehow the transmitter must know the channel transfer function or impulse response. The resulting signal $f(n)$ is subtracted from the input $a(n)$ to form the input to the mod Λ_p box.

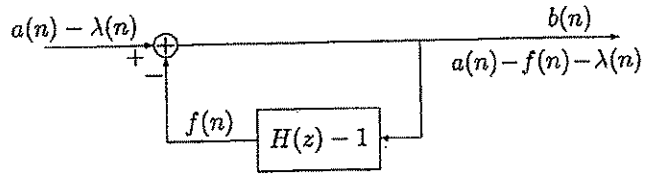


Figure 5.4. Another Equivalent Precoder Block Diagram

Another equivalent precoder block diagram is shown in Figure 5.4. This system assumes that the quantized sequence of points $\lambda(n)$ from the precoding lattice is magically known. The input $a(n) - \lambda(n)$ passes through a linear single loop feedback system. The transfer function of the feedback loop is

$$G(z) = \frac{1}{1 + [H(z) - 1]} = \frac{1}{H(z)} \tag{5.1}$$

which is the exact inverse of the channel. In Z-transform notation, the channel input is

$$B(z) = [A(z) - \Lambda(z)]\frac{1}{H(z)} \tag{5.2}$$

and the channel output is

$$R(z) = B(z)H(z) = A(z) - \Lambda(z) \tag{5.3}$$

or in the time domain

$$r(n) = b(n) * h(n) = a(n) - \lambda(n) \tag{5.4}$$

where $*$ denotes convolution.

Additive channel noise was not considered in the previous derivation. It is customary in modem performance evaluations to add any channel noise to the channel output just before the receiver. Then it is only necessary to modify (5.4) by adding a noise signal to the right-hand side.

Another method of deriving (5.4) is to observe from Figure 5.4 that

$$b(n) = a(n) - \lambda(n) - [b(n) * h(n) - b(n)] \tag{5.5}$$

so

$$b(n) * h(n) = a(n) - \lambda(n) \tag{5.6}$$

The channel output is $r(n) = b(n) * h(n)$, so on using this result in (5.6) we again find that the channel output is $r(n) = a(n) - \lambda(n)$.

Remember that the precoder input $a(n)$ is constrained to lie in a fundamental region of the precoding lattice Λ_p and that $\lambda(n)$ is an element of Λ_p. Thus, $a(n) - \lambda(n)$ must fall outside the original fundamental region whenever $\lambda(n)$ is not 0. This is not a problem because $a(n)$ can be uniquely determined from the received signal $r(n) = a(n) - \lambda(n)$ by reducing $r(n)$ modulo Λ_p to fall in the bounding fundamental region for the signal constellation. That is, $a(n)$ is the unique element in the set $\{r(n) + \lambda \,|\, \lambda \in \Lambda_p\}$ that falls in the bounding fundamental region.

Two approaches can be used to estimate $a(n)$ at the receiver when the precoder is cascaded with a trellis encoder. In the first approach, the received signal $r(n)$ can first be translated into the fundamental region of Λ_p by subtracting off elements of Λ_p as described in the previous paragraph and then a standard Viterbi decoder can be used. The second approach is based on the observation that $a(n) - \lambda(n)$ is in the same coset of Λ_c' as $a(n)$ since $\lambda(n)$ is an element of Λ_p which was restricted to be a sublattice of Λ_c'. Thus a Viterbi decoder can be used to estimate $a(n) - \lambda(n)$ directly from r(n). Then $a(n)$ can be estimated from the decoder output by translating the result into the original fundamental region of Λ_p as described in the previous paragraph.

Harashima [32] rearranges the precoder block diagram into the form shown in Figure 5.5 which explicitly shows the formation of $a(n) - \lambda(n)$ using what is called "transformation T." The signal $d(n)$ is again passed through a feedback system whose transfer function is $1/H(z)$, the reciprocal of the channel transfer function, so the signal $d(n) = a(n) - \lambda(n)$ is observed at the output of the channel.

It can be shown that if $a(n)$ is a sequence of identically distributed independent random variables that are uniformly distributed over the bounding fundamental region, then $b(n)$ is uniformly distributed over the chosen fundamental region of Λ_p contained in its Voronoi region. Therefore, the shaping gain for a constellation containing a large number of points is close to the shaping gain of the Voronoi region. In the 2D example discussed above, the Voronoi region is an $M \times M$ square so no shaping gain is achieved. When the precoder input $a(n)$ is a sequence of points from a shaped constellation and the channel impulse

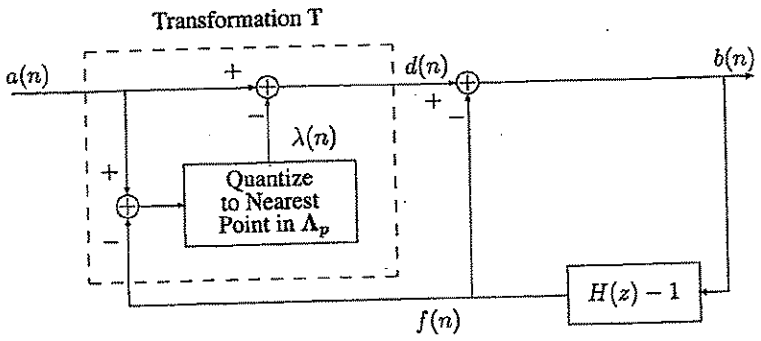


Figure 5.5. A Form of the Precoder Suggested by Harashima

response extend over several bauds, the Tomlinson precoding operation tends to destroy the desired shaping gain. Therefore, Tomlinson/Harashima precoding was not selected for the V.34 recommendation.

5.2 LTF/Motorola/GDC Precoding

Rajiv Laroia, Steven A. Tretter, and Nariman Farvardin (LTF) of the University of Maryland; Motorola Information Systems; and General DataComm at about the same time all discovered a new method for nonlinear precoding that allows constellation shaping. Laroia, Tretter, and Farvardin submitted their manuscript entitled "A Simple and Effective Precoding Scheme for Noise Whitening on Intersymbol Interference Channels" to the *IEEE Transactions on Communications* on January 2nd 1992 and it was published in 1993 [48]. It was also presented at the Princeton Conference on Information Sciences and Systems in March 1992 [47]. They also obtained a U.S. patent for this idea [46]. Motorola presented a paper entitled "A Flexible Form of Precoding for V.fast" at a TIA meeting later in January 1992 [53] and General DataComm presented a paper entitled "Distribution-Preserving Tomlinson Algorithm" at the same meeting [28].

A basic block diagram of the LTF precoder is shown in Figure 5.6 and a more detailed block diagram is shown in Figure 5.7. The input $a(n)$ to the precoder is assumed to be the output of a channel trellis encoder based on a lattice partition Λ/Λ'. The mod Λ' box is shown in more detail in Figure 5.7. As in the Tomlinson/Harashima precoder, this box quantizes its input $f(n)$ to the nearest point $q(n)$ in Λ' and outputs the quantization error $m(n)$. We will call $q(n)$ the *quantized feedback signal*. As in the Tomlinson precoder, quantization ambiguities that can result when the input is equidistant from two or more lattice points can be resolved by requiring that the quantization error

$m(n)$ fall in a fundamental region $\mathbb{R}_Q$ of Λ' contained in the Voronoi region of Λ'.

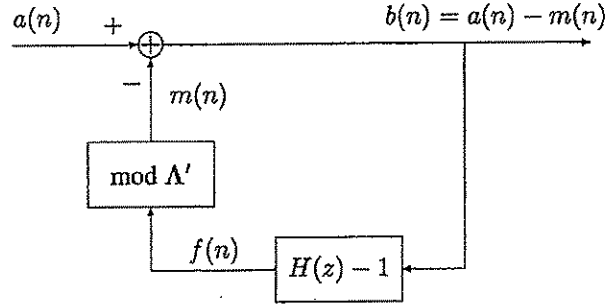


Figure 5.6. Basic Block Diagram for the LTF Precoder

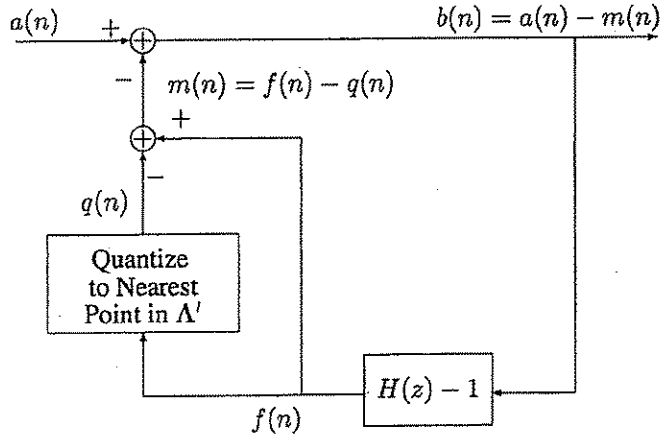


Figure 5.7. Detailed Block Diagram of LTF Precoder

At first glance, this new precoder seems very similar to the Tomlinson precoder. However, in the Tomlinson precoder the mod Λ_p box quantizes its input to the nearest point in the lattice Λ_p. The precoding lattice Λ_p is a coarse lattice which has a fundamental region that encloses the entire signal constellation. In the LTF precoder the mod Λ' box outputs a quantization error $m(n)$ that is confined to the Voronoi region of Λ'. The channel code sublattice Λ' is usually a much finer lattice than Λ_p particularly when a constellation with a large number of points is being used. Thus, the precoder output, $b(n) = a(n) - m(n)$,

is a slightly modified version of its input $a(n)$. The perturbation $m(n)$ is often called the *dither* signal. It is shown in Chapter 10 how the dither signal can be further reduced to fall in the Voronoi region of the constellation point lattice by putting the precoder and channel encoder in a feedback loop.

A mathematically equivalent form for the LTF precoder is shown in Figure 5.8. In this form, the input $a(n)+q(n)$ is passed through a single loop feedback system to give the output signal $b(n)$. This loop has the transfer function $1/H(z)$ so that the precoder output Z-transform is

$$B(z) = [A(z) + Q(z)]/H(z) \tag{5.7}$$

When $b(n)$ is applied to the channel, the resulting output is

$$R(z) = B(z)H(z) = A(z) + Q(z) \tag{5.8}$$

or in the time domain

$$r(n) = a(n) + q(n) \tag{5.9}$$

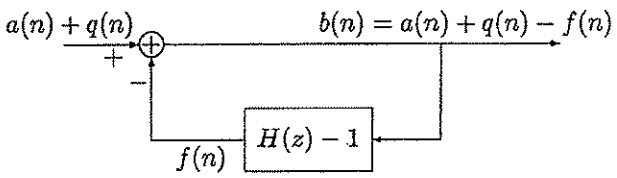


Figure 5.8. An Equivalent Form of the LTF Precoder

A block diagram of a system for recovering $a(n)$ from the received signal is shown in Figure 5.9. According to (5.9), the channel output is $r(n) = a(n) + q(n)$. Since $q(n)$ is a sequence of points from the coding sublattice Λ', the point $a(n)+q(n)$ is in the same coset of Λ' as $a(n)$ and is also a valid trellis sequence if $a(n)$ is. In practice, the channel also adds noise to $r(n)$. A Viterbi decoder for the channel trellis code can be used to estimate $a(n) + q(n)$ from the noise corrupted received signal $r(n)$. The decoder output is called $v(n)$ in Figure 5.9.

If the decoder makes no errors, $v(n) = a(n)+q(n)$, so the transmitted signal $b(n) = a(n) - m(n) = a(n) + q(n) - f(n)$ can be regenerated by passing $v(n)$ through the feedback loop shown if Figure 5.8 or, equivalently, through a filter with the transfer function $1/H(z)$ which is the block labeled "Feedback Removing Filter" in Figure 5.9. When no decoding errors have occurred, the output of this filter is $w(n) = a(n) - m(n)$. Remember that $m(n)$ is the quantization error generated in the precoder and is limited to a fundamental region $\mathbb{R}_Q$ of Λ' contained in its Voronoi region. Thus, $w(n)$ is closer to $a(n)$

than to any other point in the same coset as $a(n)$ except when $m(n)$ is on the boundary of the Voronoi region for Λ'. The coset containing $a(n)$ is the same as the coset containing the Viterbi decoder output $v(n) = a(n) + q(n)$ if no decoding errors occurred. The original transmitted sequence $a(n)$ is recovered by quantizing the output of the feedback removing filter $w(n)$ to the unique closest point in the same coset as $a(n)$ with the quantization error constrained to fall in $\mathbb{R}_Q$.

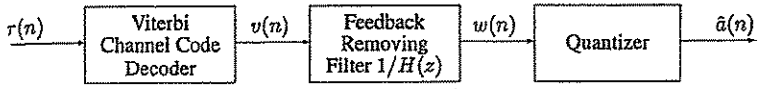


Figure 5.9. System for Recovering a(n) at the Receiver

A form of the receiver that is easier to understand and implement is shown in Figure 5.10. The feedback loop containing $H(z) - 1$ is the same as in Figure 5.8. Thus, when the Viterbi decoder makes no errors, the output $\hat{f}(n)$ of the feedback loop in Figure 5.10 is $f(n)$. The quantizer is identical to the one in the transmitter's precoder, so its output is $q(n)$ which is subtracted from $v(n) = a(n) + q(n)$ to regenerate the input $a(n)$.

The transmitted sequence $b(n)$ is the difference between the original constellation point sequence $a(n)$ and the quantization error $m(n)$. When the original constellation contains many points and the filter $H(z)$ has several taps, $m(n)$ is approximately uncorrelated with $a(n)$, so the transmitted power is approximately the sum of the powers of $a(n)$ and $m(n)$. Under these conditions, $m(n)$ is also nearly uniformly distributed over the fundamental region $\mathbb{R}_Q$ of Λ' so the average power of $m(n)$ can be approximated by the second moment of $\mathbb{R}_Q$. When the constellation for $a(n)$ contains many points, its average power will be much larger than the power of $m(n)$, so the precoding will increase the transmitted power only slightly and cause a very small reduction in any shaping gain. LTF [48] and Motorola [53] present examples showing that the loss in shaping gain is less than 0.1 dB when the 2D constellation has at least 128 points. Motorola also has an example with 5 bits per 2D symbol and the shaping loss is 0.28 dB.

An alternative form of the precoder can be developed by manipulating the block diagram in Figure 5.7 as shown in Figures 5.11 and 5.12. The change from Figure 5.7 to 5.11 is obvious. The transfer function from $V(z)$ to $B(z)$ is $1/H(z)$ and the transfer function from $V(z)$ to $F(z)$ is

$$\frac{H(z) - 1}{H(z)} = 1 - \frac{1}{H(z)} \tag{5.10}$$

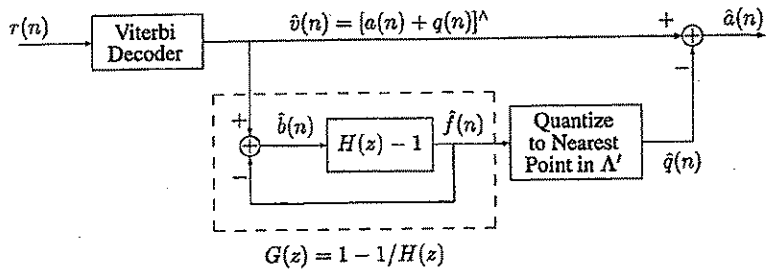


Figure 5.10. Details of System for Estimating $a(n)$ from $r(n)$

Also,

$$\frac{1}{H(z)} = 1 - \left[1 - \frac{1}{H(z)}\right] \tag{5.11}$$

Thus Figure 5.11 is mathematically equivalent to Figure 5.12. This is the form for the precoder presented in the TIA/CCITT January paper by Motorola [53].

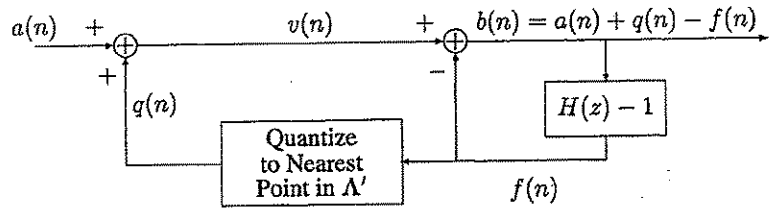


Figure 5.11. First Step to an Alternative Precoder Form

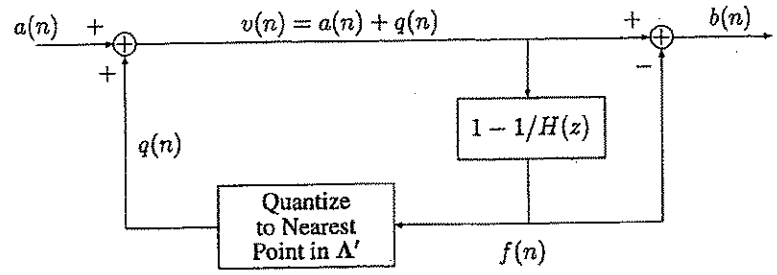


Figure 5.12. Motorola Form of the Precoder

The transfer function $H(z)$ must be minimum phase, that is, have all its zeros inside the unit circle, for the system to operate since $1/H(z)$ will be

unstable otherwise. Even when $H(z)$ is minimum phase, the impulse response corresponding to $1/H(z)$ can have a long duration if the zeros of $H(z)$ are close to the unit circle. This poses a potential for significant error propagation when the Viterbi decoder makes errors. A solution to limiting the error propagation is to approximate $C(z) = 1/H(z)$ by an FIR filter $\hat{C}(z)$ in the precoder with the form of Figure 5.12 at the transmitter and in the inverse precoder with the form of Figure 5.10 in the receiver. The error propagation is then limited to the length of the FIR approximation in the receiver. The fact that $\hat{C}(z)H(z)$ only approximates unity can be corrected, in practice, by the usual adaptive equalizer in the receiver when it is adjusted to make its output as close as possible to points in the coding lattice Λ.

EXAMPLE 5.1 LTF Precoding with the 4D Wei Code

The LTF precoder can be cascaded with the 16-state 4D Wei trellis code presented in Chapter 10. It was described there how three output bits from the Wei binary convolutional encoder select one of eight 4D subsets and a fourth bit selects a subtype within the 4D subset. The 4D subsets are translated cosets of the lattice $R\mathbf{D}_4$ which can be partitioned into $R\mathbf{D}_4 = 2\mathbf{Z}^4 \cup [2\mathbf{Z}^4 + (1,1,1,1)]$. The subtypes are the translated cosets of $2\mathbf{Z}^4$ and the corresponding translations of $2\mathbf{Z}^4 + (1,1,1,1)$. The coding sublattice for this example is $\Lambda' = R\mathbf{D}_4$. Points from this lattice consist of a pair of 2D symbols whose coordinates are all even (from $2\mathbf{Z}^4$) or a pair of 2D symbols whose coordinates are all odd (from $2\mathbf{Z}^4 + (1,1,1,1)$).

The block in Figure 5.7 that quantizes the feedback signal $f(n)$ to the nearest point in Λ' should be a 4D quantizer, not a 2D quantizer, for optimum performance with this code. The quantized pairs of 2D symbols $[q(2n), q(2n+1)]$ must be elements of the 4D coding sublattice Λ' to ensure that the channel output is a valid trellis sequence. The 4D quantization can be accomplished by combining the results of 2D quantizations. First $f(2n)$ can be quantized to the nearest points in $2\mathbf{Z}^2$ and $2\mathbf{Z}^2 + (1,1)$ and the pair of squared 2D quantization errors recorded. Then the same procedure can be performed on $f(2n+1)$. Then a squared error for points in $2\mathbf{Z}^4$ can be formed by adding the pair of 2D squared errors for the quantizations of $f(2n)$ and $f(2n+1)$ based on $2\mathbf{Z}^2$. Similarly, the squared error for points in the 4D coset $2\mathbf{Z}^4 + (1,1,1,1)$ can be formed by adding the errors for the quantizations of $f(n)$ and $f(2n+1)$ based on $2\mathbf{Z}^2 + (1,1)$. The quantizer should then choose the 4D point with the smallest squared error.

A suboptimal quantization method is based on the fact that the 4D subtypes are translated cosets of $2\mathbf{Z}^4$ which is a sublattice of $R\mathbf{D}_4$. The feedback signal $f(n)$ can be quantized on a 2D basis to the nearest point in $2\mathbf{Z}^2$ without considering 4D operations. The quantized pairs $[q(2n), q(2n+1)]$ will be elements of

$2\mathbf{Z}^4$ and also elements of the coding sublattice $R\mathbf{D}_4$. This method is not quite as good as the 4D quantization since $R\mathbf{D}_4$ is denser than $2\mathbf{Z}^4$ and the quantization error will be larger. However, it is computationally simpler.

The Wei 4D code is one of the options selected for V.34 modems. As explained in Chapter 10, additional reduction in the dither signal is achieved by embedding the precoder and channel encoder in a feedback loop. Only 2D quantizations are required in the precoder. A simple additional modification ensures that the channel output is a trellis sequence. ■

5.3 Precoding and Noise Whitening

Precoding is included in ITU-T Recommendation V.34. However, it was not included for equalizing the voiceband telephone channel. Rather, it was included to compensate for a noise whitening filter in the receiver. The block diagram for a system with noise whitening is shown in Figure 5.13. At the transmitter, a sequence of constellation points, $a(n)$, generated by a channel trellis encoder is applied to an LTF precoder designed with a filter $H(z)$ which is identical to one in the receiver. The precoder output, $b(n)$, is spectrally shaped and modulated in the conventional QAM manner [60] and sent over the telephone line. At the receiver, a conventional adaptive equalizer is adjusted so that when the precoding is bypassed, that is, $H(z) = 1$, the equalizer baseband 2D output samples $c(n)$ fall on the channel trellis code lattice points when channel noise is not present. The equalizer enhances and correlates additive channel noise, particularly when the channel amplitude response rolls off rapidly at the band edges.

The standard Viterbi decoding algorithm based on the cumulative squared Euclidean error metric is optimum for the case of uncorrelated, additive, Gaussian noise. Correlated noise causes the error rate to increase when a decoder designed for uncorrelated noise is used. One solution to this problem is to place a noise whitening filter between the adaptive equalizer and Viterbi decoder. This filter is called the *Noise Prediction Error Filter* in Figure 5.13 and is assumed to have a transfer function $H(z)$. The choice of $H(z)$ is discussed in more detail below. The intersymbol interference introduced by $H(z)$ is eliminated by using a precoder in the transmitter matched to $H(z)$.

The noise whitening filter $H(z)$ is based on the theory of linear prediction one sample into the future. Suppose $c(n)$ is a stationary random process with the power spectral density

$$S_c(z) = \sigma^2 B(z)\bar{B}(z^{-1}) \tag{5.12}$$

where $B(z)$ has the form

$$B(z) = \frac{1 + a_1 z^{-1} + \cdots + a_L z^{-L}}{1 + d_1 z^{-1} + \cdots + d_N z^{-N}} \tag{5.13}$$

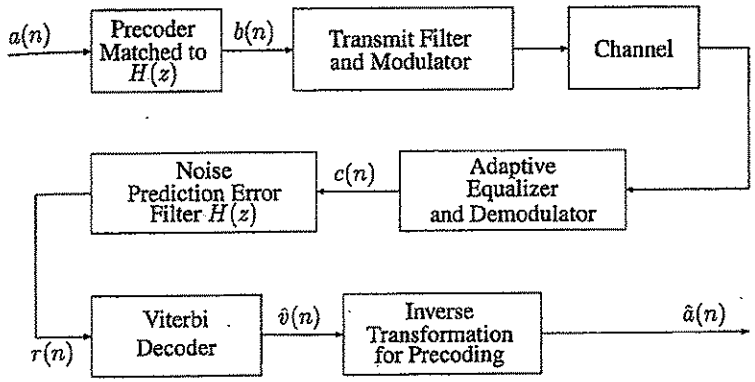


Figure 5.13. A System with Precoding and Noise Whitening

with all the poles and zeros of $B(z)$ inside the unit circle and $\bar{B}(z)$ is $B(z)$ with all its coefficients conjugated. The problem is to optimally estimate $s(n) = c(n)$ in the minimum mean-square error sense from $\{c(k)\,;\, k \le n-1\}$, that is, from all past values, by passing $c(n)$ through a filter with transfer function $P(z)$. It is shown in [59] on pages 181–182, that the optimum prediction filter transfer function is

$$P(z) = 1 - \frac{1}{B(z)} \tag{5.14}$$

The prediction error is $r(n) = c(n) - s(n)$ or in the z-domain

$$R(z) = C(z) - P(z)C(z) = [1 - P(z)]C(z) \tag{5.15}$$

Thus, the prediction error is generated by passing $c(n)$ through a filter with the transfer function

$$H(z) = 1 - P(z) = 1 - \left[1 - \frac{1}{B(z)}\right] = \frac{1}{B(z)} \tag{5.16}$$

The power spectral density for the prediction error is

$$S_r(z) = S_c(z)H(z)\bar{H}(z^{-1}) = \sigma^2 B(z)\bar{B}(z^{-1})\frac{1}{B(z)\bar{B}(z^{-1})} = \sigma^2 \tag{5.17}$$

Therefore, when the optimum prediction error filter is used, the prediction error is white (uncorrelated) with the constant power spectral density σ^2. The mean-square prediction error for the optimum filter is

$$\sigma_r^2 = \mathrm{E}\{|r(n)|^2\} = \frac{1}{2\pi}\int_{-\pi}^{\pi} S_r(e^{j\omega})\,d\omega = \sigma^2 \tag{5.18}$$

where E denotes statistical expectation. It can be shown (see [14], Chapter XII, p. 577) that the mean-square prediction error for the optimum predictor can be computed in terms of the power spectral density for $c(n)$ as

$$\sigma_r^2 = \sigma^2 = \exp\left\{\frac{1}{2\pi}\int_{-\pi}^{\pi} \log_e S_c(e^{j\omega})\, d\omega\right\} \tag{5.19}$$

From (5.17) and (5.18) it follows that

$$S_c(z) = \frac{\sigma_r^2}{H(z)\bar{H}(z)} \tag{5.20}$$

so

$$\sigma_c^2 = \mathrm{E}\{|c(n)|^2\} = \frac{1}{2\pi}\int_{-\pi}^{\pi} S_c(e^{j\omega})\, d\omega = \sigma_r^2 \frac{1}{2\pi}\int_{-\pi}^{\pi} \frac{1}{|H(e^{j\omega})|^2}\, d\omega \tag{5.21}$$

The ratio of the powers of the input and output of the prediction error filter, which is called the prediction gain, is

$$\gamma_p = \frac{\sigma_c^2}{\sigma_r^2} = \frac{1}{2\pi}\int_{-\pi}^{\pi} \frac{1}{|H(e^{j\omega})|^2}\, d\omega \tag{5.22}$$

The optimum prediction filter is an IIR filter in general. It can be approximated in the receiver by an FIR filter and the taps of this approximate filter can be adaptively adjusted using the LMS algorithm [60]. A closed form expression for the taps of the optimum finite-order predictor can be derived (see [59], Chapter 7) and an efficient iterative computational algorithm for computing the tap values called the Levinson algorithm can be used [51]. An Nth order FIR predictor is implemented by the following convolution:

$$s(n) = \hat{c}(n) = -\sum_{k=1}^{N} h_k c(n-k) \tag{5.23}$$

and the prediction error is

$$r(n) = c(n) - \hat{c}(n) = c(n) + \sum_{k=1}^{N} h_k c(n-k) \tag{5.24}$$

which is also a convolution. The prediction error filter has the transfer function

$$H(z) = 1 + \sum_{k=1}^{N} h_k z^{-k} \tag{5.25}$$

The LMS updating formula is

$$h_i(n+1) = h_i(n) + \alpha r(n)\bar{c}(n-i) \quad \text{for } i = 1, \ldots, N \tag{5.26}$$

where α is a small positive constant that controls the convergence speed and steady-state jitter.

5.3.1 The First-Order Linear Predictor

As an example, consider a first-order predictor with the transfer function $P(z) = -h_1 z^{-1}$. The predicted value is

$$s(n) = \hat{c}(n) = -h_1 c(n-1) \tag{5.27}$$

and the prediction error is

$$r(n) = c(n) - \hat{c}(n) = c(n) + h_1 c(n-1) \tag{5.28}$$

or in the z-domain

$$R(z) = C(z) + h_1 z^{-1} C(z) = [1 + h_1 z^{-1}]C(z) \tag{5.29}$$

Thus, the prediction error can be generated by passing $c(n)$ through a filter with the transfer function

$$H(z) = 1 + h_1 z^{-1} \tag{5.30}$$

The autocorrelation function for $c(n)$ can be defined as

$$R_c(k) = \mathrm{E}\{c(n+k)\bar{c}(n)\} \tag{5.31}$$

The average power in $c(n)$ is $R_c(0) = \mathrm{E}\{|c(n)|^2\}$. Let the autocorrelation function normalized by the average power be represented by the function

$$\rho_c(k) = R_c(k)/R_c(0) \tag{5.32}$$

The magnitude of the normalized autocorrelation function can be shown to be less than or equal to one for all k.

The mean-square prediction error is

$$\begin{aligned}\sigma_r^2 &= \mathrm{E}\{|r(n)|^2\} = \mathrm{E}\{[c(n) + h_1 c(n-1)]\overline{[c(n) + h_1 c(n-1)]}\} \\ &= \mathrm{E}\{|c(n)|^2\} + h_1 \mathrm{E}\{\bar{c}(n)c(n-1)\} + \bar{h}_1 \mathrm{E}\{c(n)\bar{c}(n-1)\} \\ &\quad + |h_1|^2 \mathrm{E}\{|c(n-1)|^2\}\end{aligned} \tag{5.33}$$

Using the definitions of the autocorrelation and normalized autocorrelation functions, the mean-square prediction error can be expressed as

$$\begin{aligned}\sigma_r^2 &= R_c(0) + h_1 \bar{R}_c(1) + \bar{h}_1 R_c(1) + |h_1|^2 R_c(0) \\ &= R_c(0) + R_c(0)[h_1 \bar{\rho}_c(1) + \bar{h}_1 \rho_c(1) + |h_1|^2]\end{aligned} \tag{5.34}$$

Completing the square for the term in rectangular brackets yields

$$\begin{aligned}\sigma_\tau^2 &= R_c(0) + R_c(0)[|\rho_c(1)|^2 + h_1\bar{\rho}_c(1) + \bar{h}_1\rho_c(1) + |h_1|^2 - |\rho_c(1)|^2] \\ &= R_c(0)[1 - |\rho_c(1)|^2] + R_c(0)[h_1 + \rho_c(1)][\bar{h}_1 + \bar{\rho}_c(1)] \\ &= R_c(0)[1 - |\rho_c(1)|^2] + R_c(0)|h_1 + \rho_c(1)|^2 \end{aligned} \tag{5.35}$$

The prediction mean-square error given by (5.35) is minimized by choosing h_1 as

$$h_1 = -\rho_c(1) = -R_c(1)/R_c(0) \tag{5.36}$$

so the first-order prediction error filter has the transfer function

$$H(z) = 1 - \rho_c(1)z^{-1} \tag{5.37}$$

The mean-square prediction error for this optimum choice is

$$\sigma_\tau^2 = R_c(0)[1 - |\rho_c(1)|^2] = R_c(0)[1 - |h_1|^2] \tag{5.38}$$

The prediction gain for this first-order predictor is

$$\gamma_p = \frac{R_c(0)}{\sigma_\tau^2} = \frac{1}{1 - |\rho_c(1)|^2} = \frac{1}{1 - |h_1|^2} \tag{5.39}$$

As a specific example, suppose $h_1 = 0.75$. Then the prediction gain is $10\log_{10}\gamma_p = 10\log_{10}(16/7) = 3.59$ dB. A voiceband telephone line channel with a band edge attenuation of 17 dB has a characteristic similar to this. This example illustrates that a very significant noise reduction can be achieved by a simple whitening filter. The V.34 study committee found that a third-order predictor achieved almost all the potential prediction gain for a large majority of voiceband telephone line channels.

Chapter 6

TRELLIS PRECODING

Trellis precoding is a technique that combines channel trellis coding, constellation shaping, and nonlinear precoding for channels with intersymbol interference. It is an extension of the trellis shaping technique presented in Chapter 4. Eyuboğlu and Forney first presented this extension at the CSI Workshop on Advanced Communication Techniques in May 1989 [16] and then at the 1990 Information Theory Symposium [17]. An enhanced version was published in the *IEEE Transactions on Information Theory* in 1992 [18]. They also presented the basic ideas of trellis precoding and some test results at a CCITT V.34 committee meeting in April 1991 [52] to support their (Motorola/CODEX) proposal to make it part of the recommendation. CODEX also began manufacturing a commercial modem including trellis precoding.

Many members of the V.34 committee were reluctant to accept the trellis precoding proposal because they considered it to be too complex. Shortly after this proposal, the LTF precoder discussed in Section 5.2 and the shell mapping shaping method discussed in Chapter 8 were discovered. The committee quickly chose to include a variant of LTF precoding and shell mapping because the new approach was relatively simple and achieved at least as much shaping gain.

6.1 Trellis Precoding Based on Shaping on Regions

Historically, Forney and Eyuboğlu invented trellis precoding from the point of view of lattice partitions. Shortly thereafter they realized that the same shaping and precoding gains could be achieved in a more flexible and understandable way by using the shaping on regions approach. Their later papers introduce trellis shaping and trellis precoding by first presenting the shaping and precoding on regions view followed by sections explaining the shaping and precoding using lattice partitions method. We will use this approach in this chapter.

6.1.1 The Transmitter

The block diagram of a trellis precoder based on shaping on regions is shown in Figure 6.1. This block diagram is similar to Figure 4.10 Encoder for Trellis Shaping on Regions except that a subtractor for $\mathbf{q}(n)$ and feedback loop with transfer function $1/H(z)$ have been added on the right-hand side. Filtering by $1/H(z)$ can be used to equalize a channel or compensate for a noise whitening prediction error filter in the receiver as described in Section 5.3. The subtraction of $\mathbf{q}(n)$ can be used to optimize any desired criterion, typically, to minimize the transmitted signal power in some respect.

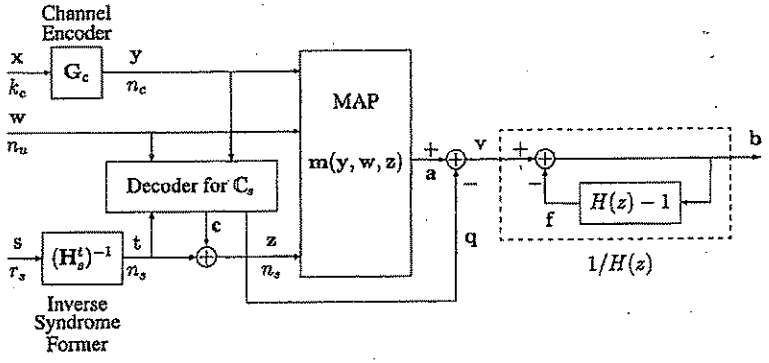


Figure 6.1. Trellis Precoder Based on Shaping on Regions

The trellis precoder has three inputs. The sequence of binary k_c-tuples $\mathbf{x}(n)$ enters the channel convolutional encoder with generator matrix $\mathbf{G}_c$ which has the sequence of binary n_c-tuples $\mathbf{y}(n)$ as its output. The sequence $\mathbf{y}(n)$ selects a sequence of cosets from the channel trellis code lattice partition Λ_c/Λ_c' which has order 2^{n_c}.

A shaping trellis code $\mathbb{C}_s$ is used which selects points from a shaping lattice Λ_s which is a sublattice of the coding sublattice Λ_c' with $|\Lambda_c'/\Lambda_s| = 2^{n_u}$ and check matrix $\mathbf{H}_s$. The input sequence of binary r_s-tuples $\mathbf{s}(n)$ is applied to an inverse syndrome former resulting in the output sequence of binary n_s-tuples $\mathbf{t}(n)$. Remember that the inverse syndrome former can be chosen so that $k_s = n_s - r_s$ of the components of $\mathbf{t}(n)$ are always zero. A binary n_s-tuple $\mathbf{c}(n)$ generated by the Decoder for $\mathbb{C}_s$ which will be explained below is added modulo 2 bitwise to $\mathbf{t}(n)$ resulting in the n_s-tuple $\mathbf{z}(n)$.

The entire constellation, $\mathbf{A}$, contains $2^{n_c+n_u+n_s}$ points selected from a convenient translation of a coding lattice Λ_c and is contained in a region $\mathbb{R}$ of N-space. This region is partitioned into 2^{n_s} subregions $\mathbb{R}^{(\mathbf{z})}$ which are selected by the shaping bits $\mathbf{z}(n)$. Each subregion contains $2^{n_c+n_u}$ points consisting of 2^{n_u} points from each of the 2^{n_c} cosets of Λ_c' in Λ_c. The binary n_u-tuple $\mathbf{w}(n)$

selects the specific point from the 2^{n_u} points in the coset of Λ_c' specified by $\mathbf{y}(n)$ in the shaping region specified by $\mathbf{z}(n)$.

The $n = n_c + n_u + n_s$ bits contained in $\mathbf{y}$, $\mathbf{w}$, and $\mathbf{z}$ select a constellation point from $\mathbf{A}$ according to an instantaneous memoryless map designated by $\mathbf{a} = \mathbf{m}(\mathbf{y}, \mathbf{w}, \mathbf{z})$. The required properties for $\mathbf{m}(\mathbf{y}, \mathbf{w}, \mathbf{z})$ were discussed in Section 4.2.1. They are:

1 $\mathbf{m}(\mathbf{y}, \mathbf{w}, \mathbf{z})$ for fixed $\mathbf{y}$ is always in the channel code coset of Λ_c' specified by $\mathbf{y}$ for all $\mathbf{w}$ and $\mathbf{z}$.

2 $\mathbf{m}(\mathbf{y}, \mathbf{w}, \mathbf{z})$ for fixed $\mathbf{z}$ is always in the shaping region $\mathbb{R}^{(\mathbf{z})}$ as $\mathbf{y}$ and $\mathbf{w}$ vary.

3 For fixed $\mathbf{y}$ and $\mathbf{z}$, the signal $\mathbf{w}$ selects one of the 2^{n_u} points from the coset specified by $\mathbf{y}$ and shaping region specified by $\mathbf{z}$.

Up to this point, the description of the system is the same as that for trellis shaping based on regions. Now the extensions for precoding will be described. First, a precoding lattice Λ_p must be chosen which is a sublattice of the shaping lattice Λ_s. Furthermore, the entire signal constellation $\mathbf{A}$ must be contained in a fundamental region $\mathbb{R}_p$ of the precoding lattice just as in Tomlinson/Harashima precoding. The ratio of the volume of $\mathbb{R}_p$ to that of a fundamental region for the channel coding lattice Λ_c is the order of the lattice partition Λ_c/Λ_p. For a fundamental region of the precoding lattice to contain the required number of constellation points, the order of this partition must be at least $|\Lambda_c/\Lambda_p| = 2^{n_c+n_u+n_s}$.

The Decoder for $\mathbb{C}_s$ chooses a path through the shaping code trellis according to a criterion that will be discussed shortly, and outputs the corresponding binary code sequence from C_s. It also outputs a sequence $\mathbf{q}(n)$ of points from the precoding lattice that are subtracted from the MAP output to form the signal

$$\mathbf{v}(n) = \mathbf{a}(n) - \mathbf{q}(n) \tag{6.1}$$

Since the mapping $\mathbf{a} = \mathbf{m}(\mathbf{y}, \mathbf{w}, \mathbf{z})$ forces $\mathbf{a}(n)$ to be a sequence from the channel trellis code and the precoding lattice Λ_p is a sublattice of the channel code sublattice Λ_c', the sequence $\mathbf{v}(n)$ must also belong to the channel trellis code and can be decoded by a Viterbi decoder for $\mathbb{C}_c$. The signal $\mathbf{v}(n)$ is then passed through a linear filter with the transfer function $1/H(z)$ which can be implemented by the feedback loop enclosed in the dotted lines in Figure 6.1. This trellis precoding technique has been proposed for use in a system like the one shown in Figure 5.13 to compensate at the transmitter for a noise-whitening prediction-error filter with transfer function $H(z)$ in the receiver.

When $H(z)$ is a prediction error filter, $h(0) = 1$ and the output $\mathbf{b}(n)$ can be expressed as

$$\mathbf{b}(n) = \mathbf{v}(n) - \mathbf{f}(n) = \mathbf{a}(n) - \mathbf{f}(n) - \mathbf{q}(n) \tag{6.2}$$

where the feedback term $\mathbf{f}(n)$ is

$$\mathbf{f}(n) = \sum_{k=1}^{\infty} h(k)\mathbf{b}(n-k) \tag{6.3}$$

In practice, $H(z)$ is chosen to be an FIR filter with a low order L so the upper limit in the sum for $\mathbf{f}(n)$ becomes L.

The decoder for $\mathbb{C}_s$ chooses a path through the shaping code trellis and a sequence $\mathbf{q}(n)$ of points from the precoding lattice to minimize the cumulative energy in the system output $\mathbf{b}(n)$. The optimum decoder would search through all possible shaping code state sequences $\mathbf{s}(n)$ and precoding lattice sequences $\mathbf{q}(n)$. The state of the entire trellis precoder consists of the state $\mathbf{s}(n)$ of the shaping encoder and the past outputs $[\mathbf{b}(n-1), \mathbf{b}(n-1), \ldots, \mathbf{b}(n-L)]$ where L is the duration of the impulse response of $H(z)$, These can be combined into the "superstate"

$$[\mathbf{s}(n), \mathbf{b}(n-1), \mathbf{b}(n-2), \ldots, \mathbf{b}(n-L)] \tag{6.4}$$

and the "supertrellis" can be searched using the Viterbi algorithm. Generally, $\mathbf{q}(n)$ can be any point in the infinite lattice Λ_p so each output $\mathbf{b}(n)$ and, consequently, the superstate can have an infinite number of values. Of course, searching a trellis with an infinite number of states is not computationally feasible. Even if $\mathbf{q}$ is limited to a finite number of points from the precoding lattice, the required computation can be impractical. Suboptimal methods for searching the supertrellis have been investigated and are called *reduced-state sequence estimation* (RSSE). Some good references for RSSE are [11], [15], [19], and [56].

The simplest member of the family of RSSE algorithms is known as *parallel decision-feedback decoding* (PDFD). In this case, the superstate is reduced to just the shaping decoder state so the past outputs are ignored as part of the state. However, past values of $\mathbf{b}(n)$ are used in computing the feedback term $\mathbf{f}(n)$. The shaping code trellis is searched with the Viterbi algorithm and the current precoding lattice point $\mathbf{q}(n)$ is selected to minimize the cumulative path metric to each current state $\mathbf{s}(n)$. Let the optimum cumulative path metric to state $\mathbf{s}$ be

$$\Gamma_{\mathbf{s}}(n) = \sum_{k=0}^{n} |\mathbf{b}(k; \mathbf{s}, n)|^2 \tag{6.5}$$

where $\mathbf{b}(k; \mathbf{s}, n)$ is the output symbol at time k on the surviving path to state $\mathbf{s}$ at time n. Each term in the sum is the branch metric for the corresponding state transition and $\mathbf{q}$. The current branch metric for a transition from state $\mathbf{s}'$ at time $n-1$ to state $\mathbf{s}$ at time n is

$$|\mathbf{b}(\mathbf{s}, n; \mathbf{s}', n-1)|^2 = |\mathbf{a}(\mathbf{s}, n; \mathbf{s}', n-1) - \mathbf{f}(n; \mathbf{s}', n-1) - \mathbf{q}(n)|^2$$

$$= |\mathbf{a}(\mathbf{s}, n; \mathbf{s}', n-1) - \sum_{i=1}^{L} h(i)\mathbf{b}(n-i; \mathbf{s}', n-1) - \mathbf{q}(n)|^2 \tag{6.6}$$

where

$\mathbf{a}(\mathbf{s}, n; \mathbf{s}', n-1)$ is the MAP output for the shaping code trellis transition from $\mathbf{s}'(n-1)$ to $\mathbf{s}(n)$,

$\mathbf{b}(\mathbf{s}, n; \mathbf{s}', n-1)$ is the corresponding precoder output,

$\mathbf{b}(n-i; \mathbf{s}', n-1)$ is the precoder output at time $n-i$ on the surviving path to state $\mathbf{s}'$ at time $n-1$,

$\mathbf{f}(n; \mathbf{s}', n-1)$ is the feedback term at time n resulting from the surviving path to state $\mathbf{s}'$ at time $n-1$.

Notice that the feedback term $\mathbf{f}(n; \mathbf{s}', n-1)$ depends only on past $\mathbf{b}$'s. Let $\mathbf{s}'$ be a state at time $n-1$ on the path to state $\mathbf{s}$ at time n. The cumulative path metric can be recursively minimized by observing that

$$\Gamma_{\mathbf{s}}(n) = \min_{\mathbf{s}', \mathbf{q}(n)} \{|\mathbf{a}(\mathbf{s}, n; \mathbf{s}', n-1) - \mathbf{f}(n; \mathbf{s}', n-1) - \mathbf{q}(n)|^2 + \Gamma_{\mathbf{s}'}(n-1)\} \tag{6.7}$$

The decoder must store path histories for the surviving shaping encoder trellis paths to each state at time $n-1$ as well as the corresponding output path histories

$$\mathbf{b}(n-L; \mathbf{s}', n-1), \mathbf{b}(n-L+1; \mathbf{s}', n-1), \ldots, \mathbf{b}(n-1; \mathbf{s}, n-1)$$

The corresponding path metrics $\Gamma_{\mathbf{s}'}(n-1)$ must also be saved. With this information, the terms

$$\mathbf{a}(\mathbf{s}, n; \mathbf{s}', n-1) - \mathbf{f}(n; \mathbf{s}', n-1)$$

in (6.7) can be evaluated for each possible transition from a previous state $\mathbf{s}'$ to the current state $\mathbf{s}$. Then the $\mathbf{q}(n)$ that minimizes (6.7) for the transition from $\mathbf{s}'$ to $\mathbf{s}$ must be the closest point in the precoding lattice Λ_p to $\mathbf{a}(\mathbf{s}, n; \mathbf{s}', n-1) - \mathbf{f}(n; \mathbf{s}', n-1)$. In other words, $\mathbf{q}(n)$ is computed by quantizing $\mathbf{a}(\mathbf{s}, n; \mathbf{s}', n-1) - \mathbf{f}(n; \mathbf{s}', n-1)$ to the nearest precoding lattice point. Then the branch metric is the squared magnitude of the quantization error and the tentative output $\mathbf{b}(\mathbf{s}, n; \mathbf{s}', n-1)$ is the quantization error itself. Mathematically, the quantization error can be described as

$$\mathbf{b}(\mathbf{s}, n; \mathbf{s}', n-1) = \mathbf{a}(\mathbf{s}, n; \mathbf{s}', n-1) - \mathbf{f}(n; \mathbf{s}', n-1) \bmod \Lambda_p \tag{6.8}$$

The procedure for PDFD described in the previous paragraph amounts to incorporating Tomlinson/Harashima precoders into the branch metric and output

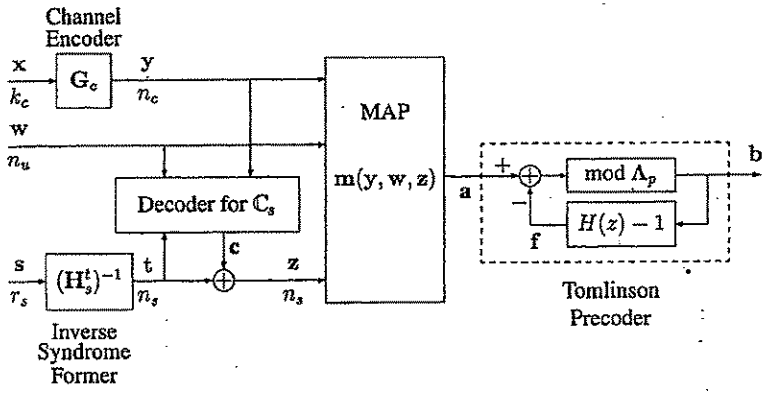


Figure 6.2. Trellis Precoder Using Parallel Decision Feedback Decoding (PDFD)

calculations with one precoder for each state. This is symbolically illustrated in Figure 6.2. The section enclosed in dotted lines and labelled "Tomlinson Precoder" represents the parallel bank of precoders.

When the input to a Tomlinson/Harashima precoder is an independent sequence uniformly distributed over a fundamental region of the precoding lattice, its output is uniformly distributed over the Voronoi region of the precoding lattice. If the Voronoi region is square, no shaping gain is achieved. The same results are approximately true when the input constellation is a discrete set of points chosen with equal likelihood from the points of a lattice contained in a fundamental region of the precoding lattice and the input constellation contains a large number of points. With trellis precoding, the output points are still confined to the Voronoi region of the precoding lattice. However, output points are no longer equally likely. Points close to the origin occur with higher likelihood than points farther from the origin. The distribution is approximately Gaussian. Thus, trellis precoding gives a shaping gain.

PDFD is suboptimal but it has been found experimentally to achieve a significant portion of the possible gain. Its complexity is about the same as a standard Viterbi decoder for the shaping code and a set of Tomlinson/Harashima precoders.

As with trellis shaping, the trellis precoder output is determined by tracing the current state with the best metric back to the end of the trellis storage array. It is important to insure that the selected path remains connected and corresponds to a sequence in the shaping code. This can be accomplished by tracing each current state back to the end of the trellis storage and assigning very large metrics to unconnected paths.

6.1.2 The Receiver

The communication channel introduces intersymbol interference and adds noise to the transmitted signal. We will assume that the "Adaptive Equalizer and Demodulator" block shown in Figure 5.13 completely removes the intersymbol interference so that its output is

$$\mathbf{d}(n) = \mathbf{b}(n) + \boldsymbol{\gamma}(n) \tag{6.9}$$

where $\boldsymbol{\gamma}(n)$ is a colored Gaussian noise sequence. Then the output of the noise prediction error filter $H(z)$ in the receiver is

$$\mathbf{r}(n) = \mathbf{v}(n) + \boldsymbol{\gamma}'(n) = \mathbf{a}(n) - \mathbf{q}(n) + \boldsymbol{\gamma}'(n) \tag{6.10}$$

where $\boldsymbol{\gamma}'(n)$ is $\boldsymbol{\gamma}(n)$ filtered by $H(z)$.

Without additive noise, the prediction error filter output is $\mathbf{v}(n) = \mathbf{a}(n) - \mathbf{q}(n)$. Remember that the original constellation points $\mathbf{a}(n)$ are taken from a set of points A confined to a region $\mathbb{R}_p$ that is a fundamental region of the precoding lattice $\boldsymbol{\Lambda}_p$ and that $\mathbf{q}(n)$ is an element of $\boldsymbol{\Lambda}_p$. Therefore, when $\mathbf{q}(n) \neq 0$ the prediction error filter output falls outside the original constellation region $\mathbb{R}_p$. However, $\mathbf{v}(n)$ is still a point in the coding lattice $\boldsymbol{\Lambda}_c$ since $\boldsymbol{\Lambda}_p$ is a sublattice of $\boldsymbol{\Lambda}_c$. Thus, $\mathbf{v}(n)$ is still a sequence from the channel trellis code if the expanded constellation is allowed.

Since $\mathbf{v}(n)$ is a sequence from the channel trellis code, it can be estimated from $\mathbf{r}(n)$ by a Viterbi decoder for the channel code. Assuming that there are no decoding errors, $\mathbf{a}(n)$ can be uniquely recovered from the sequence $\hat{\mathbf{v}}(n)$ estimated by the decoder by translating $\hat{\mathbf{v}}(n)$ by points from $\boldsymbol{\Lambda}_p$ until the result falls into the fundamental region $\mathbb{R}_p$. That is, from the set $\{\hat{\mathbf{v}}(n)+\mathbf{q}(n) \mid \mathbf{q}(n) \in \boldsymbol{\Lambda}_p\}$ select the unique point that falls in $\mathbb{R}_p$ as $\mathbf{a}(n)$.

An alternative approach to decoding is to first translate the received sequence $\mathbf{r}(n)$ by points from $\boldsymbol{\Lambda}_p$ until it falls in the fundamental region $\mathbb{R}_p$ and then perform Viterbi decoding.

Once $\mathbf{a}(n)$ has been determined, the original data bits can be found using exactly the same inverse transformation as for trellis shaping which is shown in Figure 4.2. Both the shaping code syndrome former $\mathbf{H}_s$ and the channel code inverse generator matrix $\mathbf{G}_c^{-1}$ can be chosen to be feedback free so that error propagation is limited.

6.1.3 An Example of a Trellis Precoding System

An example of a trellis precoding system for transmitting 4 information bits per 2D symbol will now be presented. The channel convolutional code is a 4-state Ungerboeck code with $k_c = 1$ and $n_c = 2$. The code selects points from the translated lattice $\mathbf{Z}^2 + (0.5, 0.5)$ and is based on the lattice partition $\mathbf{Z}^2/2\mathbf{Z}^2$ which has order 4. The encoder output $\mathbf{y}(n) = [y_1(n), y_2(n)]$ specifies

the coset of $2\mathbf{Z}^2$. These will be called the *coding cosets*. The generator matrix can be chosen to be

$$\mathbf{G}_c(D) = [1 + D^2, 1 + D + D^2] \tag{6.11}$$

An inverse generator matrix that can be used in the receiver is

$$\mathbf{G}_c^{-1}(D) = \begin{bmatrix} 1+D \\ D \end{bmatrix} \tag{6.12}$$

The shaping code is also a version of the Ungerboeck 4-state code. However, this code is based on the lattice partition $4\mathbf{Z}^2/8\mathbf{Z}^2$ which also has order 4 and code points are selected from the untranslated lattice $\Lambda_s = 4\mathbf{Z}^2$. A block diagram of the shaping encoder and corresponding trellis is shown in Figure 6.3.

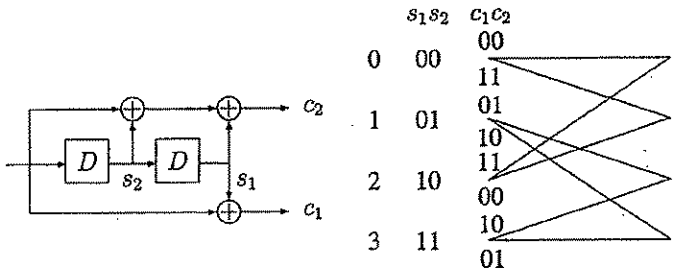


Figure 6.3. Encoder for C_s and its Trellis

The inverse syndrome former for the Ungerboeck code can be selected as

$$(\mathbf{H}_s^t)^{-1} = \left[\frac{1}{1 + D + D^2}, 0\right] \tag{6.13}$$

The output of the inverse syndrome former is the $n_s = 2$ bit vector $\mathbf{t}(n) = [t_1(n), 0]$. This sequence is combined with the output of the decoder for $\mathbb{C}_s$ to give the 2-bit vector $\mathbf{z}(n)$ which is used to select one of four shaping regions $\mathbb{R}^{(\mathbf{z})}$ which are the 4×4 squares in each of the quadrants shown in Figure 6.4. The corresponding syndrome former used in the receiver is

$$\mathbf{H}_s(D) = [1 + D + D^2, 1 + D^2] \tag{6.14}$$

The $n_u = 2$ uncoded bits $\mathbf{w}(n) = [w_1(n), w_2(n)]$ specify one of four points in each coding subset of each shaping region.

The entire constellation A before precoding contains

$$M = 2^{n_c+n_u+n_s} = 2^{2+2+2} = 64 \tag{6.15}$$

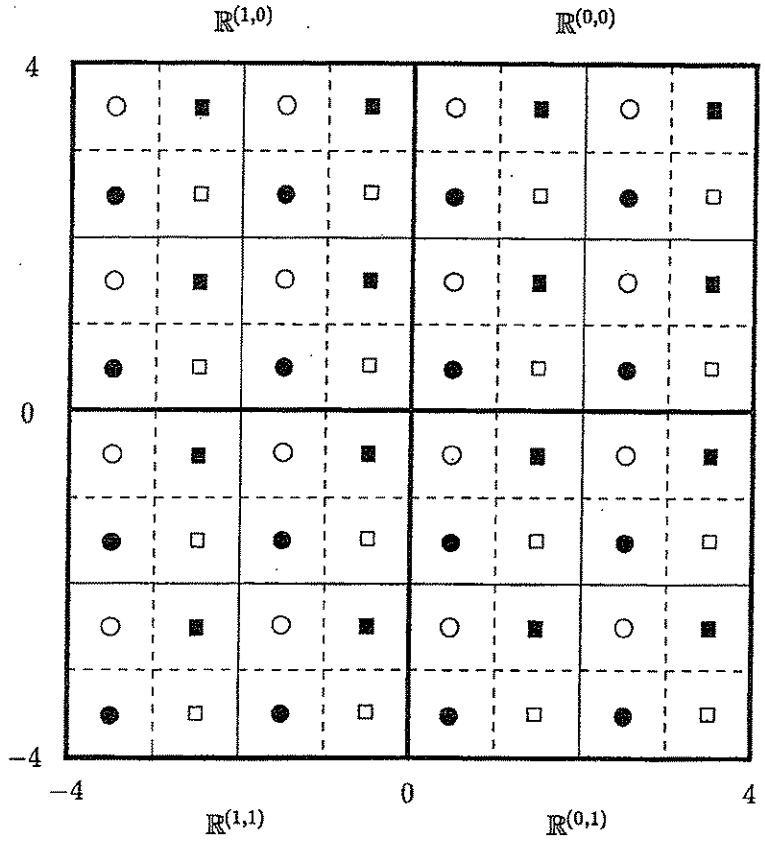


Figure 6.4. Signal Constellation Before Precoding

points. One possible constellation is shown in Figure 6.4 where points from each of the coding cosets are represented by different symbols. Notice that the entire constellation is included in the region $\{-4 < x \leq 4, -4 < y \leq 4\}$ which is a fundamental region for $8\mathbf{Z}^2$. Therefore, the precoding lattice can be selected to be $\Lambda_p = 8\mathbf{Z}^2$.

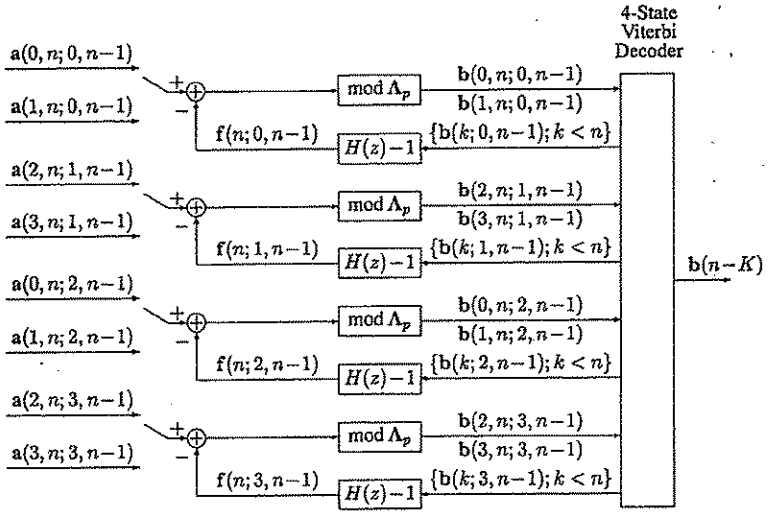


Figure 6.5. PDFD Decoder of $\mathbb{C}_s$ with Tomlinson/Harashima Precoders

A block diagram for the "Decoder for $\mathbb{C}_s$" using PDFD and incorporating a Tomlinson/Harashima precoder into each surviving path is shown in Figure 6.5. Each shaping encoder trellis state has two branches converging on it and two states diverging from it. For example, state 0 at time $n-1$ has branches leaving it to states 0 and 1 at time n with the branch symbols $\mathbf{a}(0, n; 0, n-1)$ and $\mathbf{a}(1, n; 0, n-1)$, respectively. At time n, the feedback filter $H(z) - 1$ for encoder state i must have its state set to the surviving precoder output sequence $\{\mathbf{b}(k; i, n-1);\ k < n\}$ to state i at time $n-1$. Then the two candidate precoder output symbols for transitions from state i at time $n-1$ can be generated. For example, the precoder output symbols for transitions from state 0 at time $n-1$ are labelled $\mathbf{b}(0, n; 0, n-1)$ and $\mathbf{b}(1, n; 0, n-1)$ in Figure 6.5. The 4-state Viterbi decoder can then use the cumulative metrics of the survivors to the four states at time $n-1$ and the eight candidate precoder outputs to select the survivors to each state at time n. The survivors are selected to minimize the metric $\Gamma_s(n)$ given by (6.7). For example, the survivor to state 0 at time n

corresponds to the path giving the smaller of

$$\Gamma_0(n-1) + |\mathbf{b}(0, n; 0, n-1)|^2 \text{ and } \Gamma_2(n-1) + |\mathbf{b}(0, n; 2, n-1)|^2 \quad (6.16)$$

6.2 Trellis Precoding Based on Lattice Partitions and Linear Codes

Another approach to trellis precoding is based on lattice partitions and linear shaping codes. As an example, let the shaping trellis code $\mathbb{C}_s(\Lambda_s/\Lambda'_s; C_s)$ be based on a lattice partition of the form $\Lambda_s/\Lambda'_s = L\mathbf{Z}^2/2L\mathbf{Z}^2$ for some positive integer L, the binary convolutional code C_s be Forney's form of the Ungerboeck 4-state code, and the precoding lattice be $\Lambda_p = \Lambda'_s = 2L\mathbf{Z}^2$. The partition Λ_s/Λ'_s has order 4. The value for L was selected to be 2 in the example of Section 6.1.3. A block diagram for the encoder is shown in Figures 3.11(b) and 6.3, and it was shown in Example 3.9 that the outputs $\mathbf{c}(n) = [c_1(n), c_2(n)]$ are coset representatives for the lattice partition Λ_s/Λ'_s when the components are considered to be real numbers rather than binary digits. This shaping trellis code is linear as described in Section 4.1.1.3 in the sense that

$$\mathbf{m}_s(\mathbf{t} \oplus \mathbf{c}) = \mathbf{m}_s(\mathbf{t}) \pm \mathbf{m}_s(\mathbf{c}) \mod \Lambda_s \quad (6.17)$$

where $\mathbf{m}_s(\cdot)$ is the mapping from the binary encoder output to the lattice coset representative. This means that the sums and differences of shaping trellis code sequences are also shaping trellis code sequences. According to (4.8), the MAP function can be expressed as

$$\mathbf{a} = \mathbf{m}(\mathbf{y}, \mathbf{w}, \mathbf{t} \oplus \mathbf{c}) = \lambda'_s + \mathbf{m}_c(\mathbf{y}) + \mathbf{m}_u(\mathbf{w}) + \mathbf{m}_s(\mathbf{t} \oplus \mathbf{c}) + \mathbf{d} \quad (6.18)$$

where λ'_s is an element of Λ'_s. Since the shaping code is linear, it follows that

$$\begin{aligned} \mathbf{a} &= \mathbf{m}(\mathbf{y}, \mathbf{w}, \mathbf{t} \oplus \mathbf{c}) = [\lambda'_s + \mathbf{m}_c(\mathbf{y}) + \mathbf{m}_u(\mathbf{w}) + \mathbf{m}_s(\mathbf{t}) + \mathbf{d}] - \mathbf{m}_s(\mathbf{c}) \\ &= \mathbf{c}_c - \mathbf{c}_s \end{aligned} \quad (6.19)$$

where $\mathbf{c}_c$ is a sequence in the channel trellis code and $\mathbf{c}_s$ is a sequence in the shaping trellis code.

A trellis encoder based on regions would then form

$$\mathbf{v} = \mathbf{a} - \mathbf{q} = \mathbf{c}_c - \mathbf{c}_s - \mathbf{q} = \hat{\mathbf{c}}_s \quad (6.20)$$

where $\hat{\mathbf{c}}_s$ is also a sequence in the shaping trellis code since $\mathbf{q} \in \Lambda'_s$. Let $G(z) = 1/H(z)$ and $g(n)$ be the corresponding time sequence. Then, in the time-domain, the output sequence is

$$\begin{aligned} \mathbf{b}(n) &= [\mathbf{c}_c(n) - \hat{\mathbf{c}}_s(n)] * g(n) \\ &= \mathbf{c}_c(n) * g(n) - \hat{\mathbf{c}}_s(n) * g(n) \end{aligned} \quad (6.21)$$

where $*$ designates convolution.

Let $\mathbb{C}_s' = \mathbb{C}_s * g$ be the trellis code consisting of the set of sequences from the original shaping trellis code $\mathbb{C}_s$ filtered by $G(z) = 1/H(z)$. Then trellis shaping and precoding can be achieved by searching for the sequence $\mathbf{c}_s'(n)$ in $\mathbb{C}_s'$ that minimizes the energy in the sequence $\mathbf{b}(n)$ in (6.20). A block diagram for this scheme is shown in Figure 6.6.

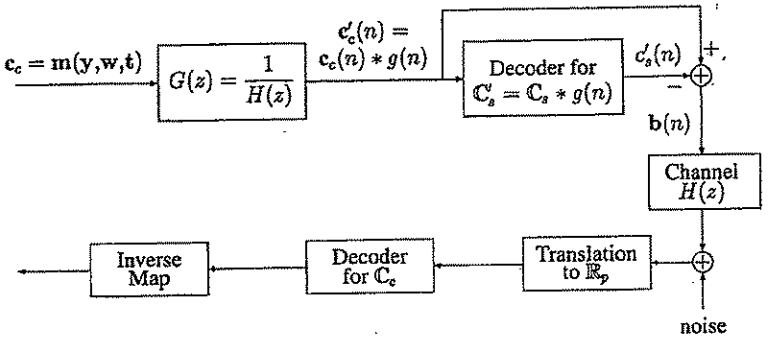


Figure 6.6. Trellis Precoding with Linear Shaping Trellis Code

The transmitted sequence $\mathbf{b}(n)$ lies in the Voronoi region of the filtered trellis code $\mathbb{C}_s'$. Thus, the shaping gain depends on both the shaping code and the channel response $H(z)$.

6.3 Experimental Performance Results

Formulas for the shaping gain as a function of the system parameters have not yet been found because of the system complexity. Eyuboğlu and Forney [18] have presented various experimental results using the Ungerboeck 4-state code showing shaping gains from 0.6 to 0.9 dB with reasonable delay and complexity. Figure 6.7 suggests the type of experimental results they observed with PDFD. It was generated by roughly measuring the values on the graphs of Figure 11 in their paper. They show that about 0.1 dB more shaping gain can be achieved by using a more complicated RSSE decoder.

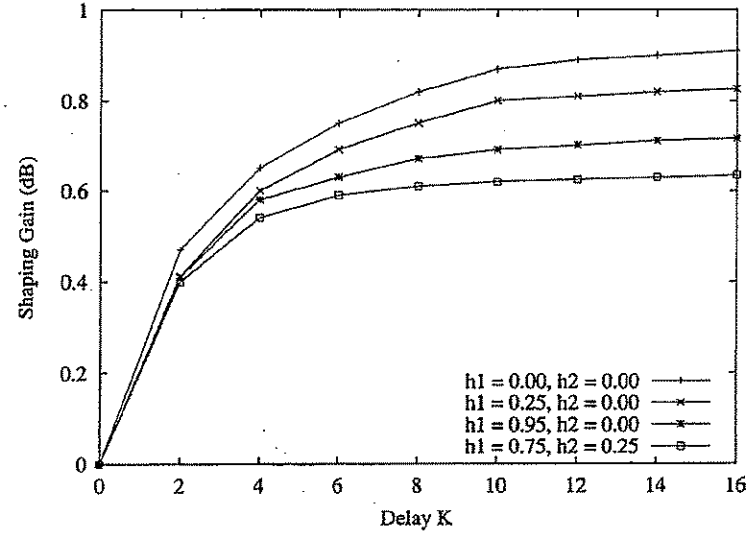


Figure 6.7. Shaping Gain with PDFB Decoding for the 4-State Ungerboeck Code with the Channel $h(D) = 1 + h_1 D + h_2 D^2$ (estimated from Fig. 11. of [18])

Chapter 7

MAPPING DATA TO CHANNEL SYMBOL FRAMES BY A MODULUS ENCODER

During the ITU-T V.34 committee deliberations, Motorola Information Systems initially advocated including trellis shaping [23] and later trellis precoding [18] in Recommendation V.34. Many committee participants were not convinced this was the best approach. They felt it was conceptually and computationally too complex. AT&T proposed an alternative called the "fractional bit rate modulus converter" coupled with the Wei 16-state 4D code [1]. Shortly after that, Laroia [45, 48] presented papers to the committee suggesting the use of shell mapping and a new precoding scheme. These are discussed in detail in Chapters 8 and 5, respectively. The committee members decided that the shell mapping and precoding method had reasonable computational complexity and good performance, and the Motorola and AT&T proposals were dropped in favor of these new proposals. Trellis shaping, trellis precoding, and the fractional bit rate modulus converter are discussed in this book for historical completeness and because they are theoretically interesting techniques that might be worth considering for future communication systems. The AT&T modulus converter method was more recently generalized and adopted for use in ITU-T Recommendations V.90 [35] and V.92 [36] for PCM transmission and the modifications are described in Section 7.2.

One goal of the V.34 and V.90 ITU-T study committees in designing these standards was to have the modems achieve the highest possible data rate consistent with the channel characteristics and a desired bit error probability. This meant that the modems should be able to transmit at a reasonably large number of data rates and, in the V.34 case, several different symbol rates. For example, V.34 modems can transmit at data rates range from 2400 up to 33600 bits per second in increments of 2400 bits per second and can use six different symbol rates. V.90 modems can transmit at data rates varying from 28000 to 56000 bits per second in increments of 8000/6 bps. Achieving a wide variety

of data rates requires transmitting a fractional number of bits per channel symbol. The modulus converter and shell mapping methods both easily allow this fractional bit rate transmission. The modulus converter proposed for V.34 uses 2D constellation points with equal likelihood and achieves very little shaping gain, just that of a circle over a square in two dimensions which is about 0.20 dB. The shell mapper selected for V.34 achieves a shaping gain close to that of a 16-dimensional sphere and the points in the 2D constituent constellations occur with close to a 2D Gaussian density centered on the origin when the constellation expansion option is selected.

7.1 The AT&T Fractional Bit Rate Modulus Converter

The AT&T modulus converter was designed to work with the 16-state 4D Wei trellis code described in Chapter 10. The 2D constituent constellation points are selected with almost equal likelihood from a subset of $2\mathbf{Z}^2 + (1,1)$ contained in a nearly circular region about the origin. The 4D points consist of pairs of 2D points. The number of points in the 2D constellation increases with the desired data rate for a given symbol rate. A positive integer index is assigned to each constellation point starting with the index 0 for the point $(1,1)$. Indexes are assigned so that a point with a larger index has a power greater than or equal to the power of one with a smaller index. A small section of the 2D constellation with indexes is shown in Figure 7.1. Notice that the constellation contains four types of points – open circles, filled squares, open squares, and filled circles. The open circles are a subset of $4\mathbf{Z}^2 + (1,1)$, the filled squares $4\mathbf{Z}^2 + (1,2)$, the open squares $4\mathbf{Z}^2 + (3,3)$, and the filled circles $4\mathbf{Z}^2 + (3,1)$. These represent the cosets of $4\mathbf{Z}^2$ in $2\mathbf{Z}^2$ translated by $(1,1)$. The open circles have indexes of the form $4n$ and will be called subset 0. The filled squares have indexes of the form $4n+1$ and will be called subset 1. The open squares have indexes of the form $4n+2$ and will be called subset 2. The filled circles have indexes of the form $4n+3$ and will be called subset 3. Clockwise rotations of 90° transform points through the following sequence: open circles to filled squares to open squares to filled circles to open circles or 0 to 1 to 2 to 3 to 0.

The modulus converter blocks consecutive input data bits into frames of b bits which in conjunction with the Wei encoder are mapped into frames of B 2D channel symbols. Since each 4D symbol consists of a pair of 2D symbols, B must be an even integer. The Wei encoder uses 3 data bits per 4D symbol to generate a parity check bit and the resulting block of four bits is used to select a pair of 2D subsets. Therefore, $3B/2$ bits per frame are used by the Wei encoder and the number of remaining uncoded bits per frame is $b - 1.5B$. The uncoded bits are used to select points in the subsets.

Each subset is required to have the same number of points, say, m. This helps make the code transparent to 90° rotations. The subset size m is called the *modulus*. The complete 2D constituent constellation contains $4m$ points. A

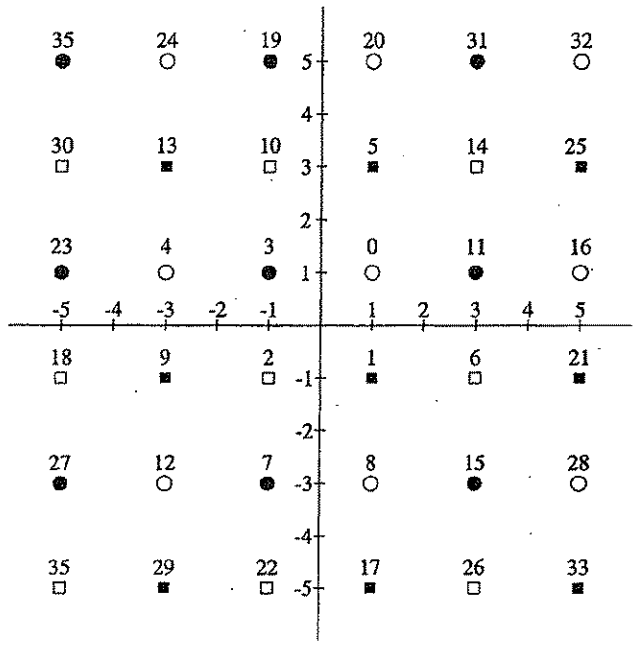


Figure 7.1. A Portion of the 2D Constellation for the AT&T Modulus Converter

lower bound on m will now be derived. The $b - 1.5B$ uncoded bits per frame select one out of a possible $2^{b-1.5B}$ sequences of subset points. Each 2D point can be selected from m points, so there are m^B possible point sequences. Thus m must satisfy the following inequality:

$$m^B \geq 2^{b-1.5B} \quad \text{or} \quad m \geq 2^{(b/B)-1.5} \tag{7.1}$$

Let f_0 be the number of frames per second. Then

$$\frac{b}{B} = \frac{f_0 b}{f_0 B} = \frac{\text{data bit rate}}{\text{2D symbol rate}} \tag{7.2}$$

So another form of the bound for m is

$$m \geq 2^{(\text{data bit rate/2D symbol rate})-1.5} \tag{7.3}$$

The modulus m can be chosen as the smallest integer satisfying the bound to get the smallest possible 2D constellation. Table 7.1 shows a variety of values for b/B bits per symbol, m points per 2D constellation, data bit rate, and symbol rate.

Table 7.1. Some Combinations of b/B, m, and Symbol Rate

Bit Rate	Symbol Rate (symbols/second)									
	2400		2743		2800		3000		3200	
(bits/s)	b/B	m	b/B	m	b/B	m	b/B	m	b/B	m
28000	24/2	1449	21/2	512	144/14	442	96/10	275	18/2	182
26400	22/2	725	77/8	280	132/14	244	88/10	158	33/4	108
24000	20/2	363	35/4	153	120/14	135	16/2	91	15/2	64
21600	18/2	182	63/8	83	108/14	75	72/10	52	27/4	39
19200	16/2	91	14/2	46	96/14	41	64/10	30	12/2	23
16800	14/2	46	49/8	25	12/2	23	56/10	18	21/4	14
14400	12/2	23	21/4	14	72/14	13	48/10	10	9/2	8
12000	10/2	12	35/8	8	60/14	7	8/2	6	15/4	5
9600	8/2	6	7/2	4	48/14	4	32/10	4	6/2	3
7200	6/2	3	21/8	3	36/14	3	24/10	2	9/4	2
4800	4/2	2	7/4	2	24/14	2	16/10	2	6/4	1

The method for selecting the constellation points will now be explained. Of the b input data bits per frame, $1.5B$ bits are used by the Wei encoder. Let the remaining $b - 1.5B$ uncoded bits be the binary representation for the decimal number n. When the modulus m satisfies the inequality (7.1), n must be in the range $[0, m^B - 1]$. Thus, n can be expressed in radix m form as

$$n = i_{B-1}m^{B-1} + i_{B-2}m^{B-2} + \cdots + i_1 m + i_0 \tag{7.4}$$

where $0 \le i_j \le m - 1$ for $j = 0, \ldots, B - 1$.

The i's can be found by repeated division. If n is divided by m, the quotient is

$$n_1 = i_{B-1}m^{B-2} + i_{B-2}m^{B-3} + \cdots + i_1 \tag{7.5}$$

and the remainder is $r_1 = i_0$. Then n_1 can be divided by m to give the quotient n_2 and remainder $r_2 = i_1$. This division process can be repeated B times to find all the i's.

The points within the subsets of the frame of B 2D symbols are selected by the i's. The first symbol in time is selected by i_{B-1} and the last symbol by i_0. The 2D subsets are selected by the outputs of the Wei encoder. Remember that the subsets are called 0, 1, 2, and 3. Let i_j be a radix m expansion coefficient and s be the corresponding subset number. Then, the index of the 2D constellation point is

$$q = 4i_j + s \tag{7.6}$$

The x and y coordinates of the point can be read from a table corresponding to an expanded version of Figure 7.1. The transmitter operations are illustrated in Figure 7.2.

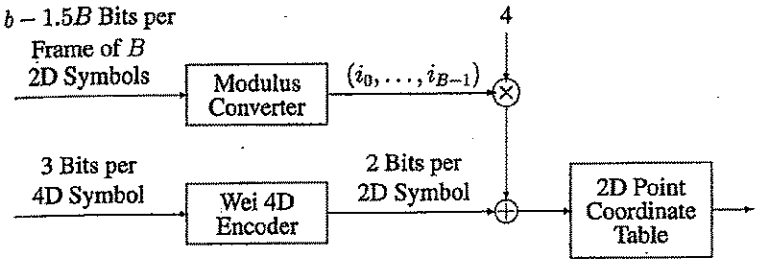


Figure 7.2. The AT&T Modulus Converter Transmitter

At the receiver, the observed sequence of constellation points is operated on by a Viterbi trellis decoder. A frame of B point indexes is collected from the decoder output and then the frame of b input bits is determined by computing n. Rather than using (7.4) for this computation, it is more efficient to compute n using the rule

$$n = (\cdots((i_{B-1}m + i_{B-2})m + i_{B-3})m + \cdots + i_1)m) + i_0 \tag{7.7}$$

In words, the first step is to multiply the first transmitted expansion coefficient i_{B-1} by m. Then i_{B-2} is added to the product and the resulting sum is multiplied by m, etc.

EXAMPLE 7.1

Suppose a data rate of 19.2 kbps is desired using a symbol rate of 3200 baud. According to (7.1)

$$m \ge 2^{(19200/3200)-1.5} = 22.63 \tag{7.8}$$

Therefore, we can use $m = 23$. The entire 2D constellation has $4m = 92$ points. According to Table 7.1, the required frame length is $B = 2$ 2D symbols with $b = 12$ data bits. Three of the data bits are used by the Wei encoder and the remaining 9 bits go to the modulus 23 converter. Notice that there are 512 possibilities for 9-bit sequences which is less than $23^2 = 529$.

Let the 9 uncoded bits be the binary representation of the number n. Then dividing n by $m = 23$ results in i_0 as the remainder and i_1 as the quotient. In the receiver, the original data bits are computed as $n = i_1 m + i_0$. ■

7.2 The V.90 Modulus Encoder

Recommendation V.90 was created to take advantage of the evolution of voiceband signal transmission inside the public switched telephone network (PSTN) from analog transmission using frequency division multiplexing (FDM) to digital transmission using pulse code modulation (PCM) and time division multiplexing (TDM). The vast majority of transmission in the PSTN now uses the digital format. The connection from a telephone in a home to a local telephone office is still primarily an analog one and consists of a twisted pair of copper wires. Signal flow in the transmit and receive directions is separated by circuits called hybrids. This connection is known as a *local loop* and the home telephone user is called a *client*. At the local office, the analog signal from the client is applied to a PCM codec which passes its input through a lowpass filter with about a 4 kHz cut-off frequency, samples the signal at an 8 kHz rate, and converts the samples to 8-bit codes using the nonlinear mu-law or A-law companding functions. The resulting 64 kbps digital data stream is multiplexed with streams from other users and routed through the internal telephone network. In the reverse direction, 8-bit PCM codes for a connection arrive at the local office through the internal digital telephone network and are applied to the digital-to-analog converter (DAC) in the codec at the rate of 8,000 samples per second. The codec converts each code to one of 256 voltage levels and passes the resulting staircase waveform through a lowpass filter with a 4 kHz cut-off frequency. The output of the filter is connected to the twisted pair through a hybrid. It is now possible for customers like internet service providers (ISP) to lease direct digital connections to the telephone network and send 8-bit PCM codes directly to a codec at the local office of a client.

The data rate of a typical analog modem like a V.34 modem is limited by the quantization noise introduced by the codec at the local office in the upstream direction, that is, from the client to the local office. It can be shown that a fully loaded μ-law or A-law digitizer limits the signal-to-noise ratio to about 38 dB. Other additive noise sources for the twisted pair are often much below this level. People suddenly realized that with a direct digital connection to the codec DAC it should be possible to transmit at nearly 64 kbps in the downstream direction, that is, from a server to a client, by sending the data as 8-bit codes at the rate of 8000 codes per second. The codes are represented by a set of 256 voltage levels applied to the local loop by the codec. No quantization noise is added by the codec and the client's receiver at the other end of the loop can use a linear adaptive equalizer to compensate for any non-ideal frequency response characteristics of the local loop. In practice, the rate must be somewhat less theoretically because of additive noise in the loop and imperfect equalization. In addition, the telephone plant sometimes uses the least significant bit of some of the PCM codes for signaling purposes which is called *robbed bit signaling* (RBS) and digital pads are sometimes used. Yet another limit on the data rate is

imposed by an arcane FCC rule limiting the power that can be applied to a local loop. The V.90 recommendation specifies data rates ranging from 28000 bps to 56000 bps in increments of 8000/6 bps. However, the FCC power regulation actually limits the maximum rate to about 53000 bps.

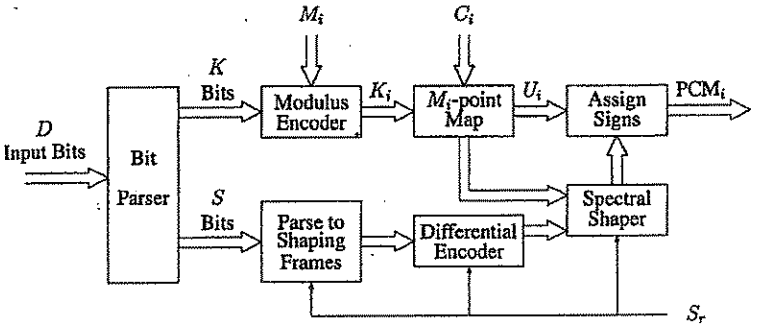


Figure 7.3. V.90 Downstream PCM Encoder

A generalization of the AT&T fractional bit rate modulus converter was selected for Recommendation V.90 to allow for a wide variety of downstream data rates. A block diagram of the encoding scheme is shown in Figure 7.3. The encoder takes in frames of D data bits. Each data frame has a duration of six PCM samples or 6/8000 seconds. Therefore, frames are transmitted at the rate of 8000/6 frames/sec. The Bit Parser sends the first S bits in the frame to a spectral shaping system which generates a frame of six sign bits. The value for S can range from 3 to 6. The spectral shaper internally generates $S_r = 6 - S$ redundant bits. Spectral shaping was included in Recommendation V.90 to account for the fact that signals are transformer coupled so the channel has a null around zero Hz and also the codec lowpass filter causes a channel spectral null around 4 kHz. See Recommendation V.90 [35] for details of the spectral shaping system. The remaining $K = D - S$ bits in the frame are passed to the Modulus Encoder which generates a frame of six integers, each corresponding to a positive PCM code level. The six sign bits are attached to the six modulus encoder outputs to form a frame of six PCM codes to be transmitted over the digital network to the codec in the client's local office. Since $D = K + S$, the data transmission rate is $R = (K + S)8000/6$ bps.

During the handshaking procedure, the client modem measures the local loop signal-to-noise ratio, and determines if and where robbed bit signaling is present. It then decides on an appropriate data rate and designs a signal constellation consisting of a subset of the positive PCM levels for each of the six samples in a frame. Each of the six constellations can be different. For example, constellations for positions corresponding to robbed bit locations

might have about half the number of levels with points twice as far apart as for other positions. The designed constellations are telemetered to the server modem during the handshake procedure. Recommendation V.90 does not say how to design the constellations. This is left up to the manufactures, most likely, because V.90 committee members who knew how to do this wanted to make it hard for outsiders to make these modems. The numbers of points in each constellation are called the *mapping moduli* and designated by M_0 to M_5 in Recommendation V.90.

The modulus encoder takes in frames of K bits. Therefore, it must be able to generate 2^K distinct frames of six samples. The number of distinct frames that can be generated for a set of six mapping moduli is $\prod_{i=0}^{5} M_i$, so they moduli must satisfy the inequality

$$2^K \leq \prod_{i=0}^{5} M_i \tag{7.9}$$

Let the row vector $(b_{K-1}, \ldots, b_1, b_0)$ be a frame of K modulus encoder input bits and let

$$R_0 = \sum_{n=0}^{K-1} b_n 2^n \tag{7.10}$$

Any integer R_0 in the set $\{0, \ldots, \prod_{i=0}^{5} M_i\}$ can be expressed in the form

$$\begin{aligned} R_0 \;=\; & K_5(M_4M_3M_2M_1M_0) + K_4(M_3M_2M_1M_0) \\ & + K_3(M_2M_1M_0) + K_2(M_1M_0) + K_1M_0 + K_0 \end{aligned} \tag{7.11}$$

where

$$0 \leq K_i \leq M_i - 1 \quad \text{for } i = 0, \ldots, 5 \tag{7.12}$$

Let $\text{mod}(x, y)$ be the remainder when x is divided by y. Then the first mapping index K_0 is the remainder

$$K_0 = \text{mod}(R_0, M_0) \tag{7.13}$$

and the quotient is

$$\begin{aligned} R_1 \;=\; & K_5(M_4M_3M_2M_1) + K_4(M_3M_2M_1) + K_3(M_2M_1) \\ & + K_2M_1 + K_1 \end{aligned} \tag{7.14}$$

Next, R_1 can be divided by M_1 to get the remainder K_1 and quotient R_2, and so on. Thus, the $K's$ can be found starting with $i = 0$ by the following computations:

$$K_i = \text{mod}(R_i, M_i) \quad \text{for } i = 0, \ldots, 5 \tag{7.15}$$

with

$$R_{i+1} = (R_i - K_i)/M_i \tag{7.16}$$

The mapping index K_i is used to select a constellation point from the constellation designed for sample frame position i which has M_i points. These points are all positive. The ith bit from the spectral shaper is used to select the polarity of the sample and the resulting 8-bit PCM code is sent to the codec.

V.90 modems use the V.34 transmission scheme in the upstream direction from the client to the server. After the success of the V.90 PCM downstream scheme, committee members decided to study PCM transmission in the upstream direction and this led to Recommendation V.92 [36]. The idea was that if the client could transmit analog samples at an 8 kHz rate and these samples could be made to arrive at the input to the analog-to-digital converter in the central office codec at the ideal codec PCM levels, there would be no quantization noise. Then the output PCM codes of the codec could be sent through the digital network to the server receiver. Upstream PCM transmission has turned out to be much more difficult than the V.90 downstream PCM transmission. First, a local loop equalization filter must be computed by the receiver at the server during a training period and sent back to the client transmitter to be used as a pre-equalizer. Then this equalizer can not be adaptively updated conveniently during normal data transmission. On the other hand, the client's receiver can contain an equalizer that is continuously adapted. Another difficulty is that the client modem must lock its transmit clock to the 8 kHz downstream PCM code rate. Any transmit clock jitter will degrade the pre-equalized signal that arrives at the codec. Still another problem is the fact that the hybrid at the central office is not perfect so some of the downstream transmitted signal from the codec echoes back to the analog input of the codec. At the codec, this echo moves the samples received from the client modem away from the ideal PCM levels and introduces quantization noise again. The server modem has no access to the local loop side of the codec and can only perform a suboptimal echo cancellation on its digital side of the codec. The V.92 recommendation was approved in November 2000, but very few servers currently (September 2001) have deployed the PCM upstream option.

Chapter 8

CONSTELLATION SHAPING BY SHELL MAPPING

A technique know as *shell mapping* is specified in the ITU-T Recommendation V.34 [34] for constellation shaping. This method achieves higher shaping gains with comparable or less computational complexity than the methods of trellis shaping [23] and trellis precoding [18] which had been proposed by Motorola Information Systems in the committee deliberations prior to the introduction of shell mapping. Shell mapping can be easily combined with trellis channel coding. Constraints on the constellation expansion ratio (CER_s) and peak-to-average ratio (PAR) in two dimensions are easy to include. Shell mapping achieves shaping gains close to that of an N-sphere if there are no PAR or CER_s constraints. Also, LTF precoders [46, 48] can be cascaded with shell mapped constellations to perform channel equalization at the transmitter without destroying shaping gain.

The initial use of shell mapping appears to be in a commercial modem manufactured by ESE [42] of Canada to map binary data blocks to 24-dimensional constellation points selected from the Leech lattice. Shell mapping was also suggested in a Ph.D. thesis by Frank Robert Kschischang [41] at the University of Toronto and he also thoroughly analyzed the problem of PAR and CER_s constraints using multidimensional polydisks. Khandani and Kabal [39, 40] also independently studied the properties of truncated polydisks but did not give them a name and they discuss some constellation addressing schemes that are not the same as the shell mapping method proposed for V.34. Independently, R. Laroia, N. Farvardin, and S. Tretter [44, 45, 46] at the University of Maryland discovered the same shell mapping and truncated polydisk ideas of Kschischang which they called structured vector quantizer (SVQ) shaping. In addition, they propose grouping the shells into rings as suggested by A.R. Calderbank and L.H. Ozarow [8] to simplify the mapping complexity yet retain most of the possible shaping gain. In May 1992 Motorola Information Systems

[54] (CODEX) presented a paper to the CCITT V.fast committee also proposing the use of shell mapping and rings to achieve shaping gain relatively easily and stated that "Trellis shaping is no longer required." At this point the committee reached a consensus to accept shell mapping as the constellation shaping method rather than more complicated methods previously under consideration.

8.1 General System Description

The block diagram for a system using shell mapping for constellation shaping is shown in Figure 8.1. The diagram shows the Wei 4D trellis encoder as an example, but any suitable encoder can be used. As described in Chapters 9 and 10, the points in the transmitted 2D constellation are a subset of $2\mathbf{Z}^2+(1,1)$ and this constellation is partitioned into four 2D subsets. The Wei encoder takes in three bits every 4D symbol and generates four output bits per 4D symbol. The first and second pairs of output bits specify the pair of 2D symbols that form the 4D subset selected by the Wei encoder.

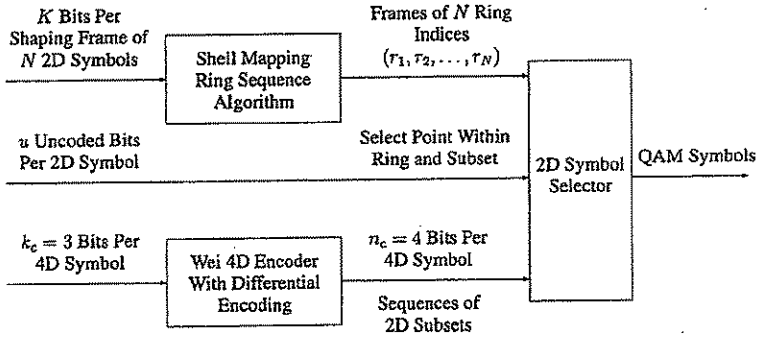


Figure 8.1. Constellation Shaping with Shell Mapping

The 2D constellation is partitioned into M "rings" so that each ring contains the same number of points according to the approach of Calderbank and Ozarow [8]. Furthermore, if the transmitter accepts u uncoded bits per 2D symbol, each ring must contain 2^u points from each of the four 2D subsets. Thus the total number of points in the 2D constellation is

$$L = 4M2^u = M2^{u+2} \tag{8.1}$$

The ring closest to the origin should contain the 2^{u+2} constellation points of least energy. The next ring from the origin should contain the next 2^{u+2} points in order of energy, and so on.

The *shell mapping* algorithm takes in successive blocks of K data bits and generates frames of ring indexes of length N. Each element in a frame corresponds to a 2D symbol interval. A total of 2^K distinct frames is required.

The basic idea behind shell mapping is that the M^N possible ring frames are arranged in an ordered list with lower weight frames appearing closer to the beginning of the list. As explained in Section 8.2, the frame weights are assigned to give a measure of the energy of the frame of N 2D symbols corresponding to the sequence of ring indexes in the ring frame. There may be many frames with the same weight and the set of them is called a shell. In Section 8.3, we will see how to lexicographically order the frames. The frame at the beginning of the list will be labeled or indexed by 0 and the one at the end by M^N-1. The 2^K lowest weight frames are used as the set of allowable shell sequences. Encoding is performed by using the binary K-tuples of shaping bits as the indexes of the elements in the ordered list. Decoding is performed by observing that the index of a ring frame is the number of frames below it on the list. We will see how to achieve reasonable table storage requirements and computational complexity by a "divide and conquer" approach by selecting N to be a power of 2 and iterating from 2 to 4 to 8, ..., to N dimensions. The implementation challenge is to find algorithms to efficiently assign shaping K-tuples to ring frames for encoding, and to assign ring frames to binary K-tuples for decoding. The shell mapping algorithm maps blocks of K bits to the 2^K least weight sequences of N rings out of the M^N possible sequences of N rings.

The constellation size must be expanded to achieve shaping gain. The ITU-T V.34 study committee chose to limit the constellation expansion ratio to 1.25 and chose a frame size of $N = 8$ 2D symbols. According to Figure 2.5, about 1.3 dB of shaping gain can be obtained with this expansion ratio by optimal shaping. However, numerical computations have shown that with a CER of 1.25, only 0.5 to 0.7 dB of shaping gain can be achieved with shell mapping but with reasonable PAR. Shaping gains of 0.6–0.8 dB can be achieved with CER = 1.5 and PAR around 3.5. With a CER of 1.5, the constellation shape is close to a 16-dimensional sphere which has a shaping gain of about 1 dB as can be seen in Figure 2.3.

A relationship between the number of rings, M, and the number of shaping bits, K, will now be derived. The K shaping bits specify 2^K sequences of N rings. Each 2D constellation is partitioned into M rings, so there are M^N possible ring sequences. Therefore, it is necessary that

$$2^K \le M^N \quad \text{or} \quad 2^{K/N} \le M \tag{8.2}$$

Each shaping frame, the transmitter accepts Nu uncoded bits, K shaping bits, and $3(N/2)$ bits that are converted to $4(N/2)$ bits by the Wei encoder. (The Wei encoder uses three input bits per 4D symbol to generates one check bit per 4D symbol. Its output consists of the three input bits and the check bit each 4D symbol.) Thus the total number of bits for constellation point selection per shaping block is

$$B = Nu + K + 4(N/2) = N(u+2) + K \tag{8.3}$$

In an unshaped system this requires a 2D constellation of size

$$[2^{N(u+2)+K}]^{1/N} = 2^{u+2+\frac{K}{N}} \tag{8.4}$$

According to (8.1), the number of points in the 2D constellation for the shell mapped system is $L = M2^{u+2}$ so the constellation expansion ratio is

$$\text{CER} = \frac{M2^{u+2}}{2^{u+2+\frac{K}{N}}} = M2^{-K/N} \tag{8.5}$$

If CER is required to be no greater than 1.25, this upper bounds M by

$$M \leq 1.25 \times 2^{K/N} \tag{8.6}$$

Combining (8.2) and (8.6), we see that M must be limited to the range

$$2^{K/N} \leq M \leq 1.25 \times 2^{K/N} \tag{8.7}$$

The selection of the number of rings, M, allows a tradeoff between constellation expansion and shaping gain. Using the smallest M gives the smallest constellation expansion and smallest shaping gain, while the largest M gives the largest constellation expansion and shaping gain.

Some additional formulas relating M, L, and K will now be derived. The number of shaping bits, K, can be divided by the shaping block length, N, to give a quotient, p, and remainder, k. Thus K can be expressed as

$$K = Np + k \quad \text{with } 0 \leq k \leq N-1 \tag{8.8}$$

Substituting (8.8) into (8.3) gives

$$B = N(u+2) + (Np+k) = N(u+2+p) + k \tag{8.9}$$

$$= nN + k \tag{8.10}$$

where

$$n = u + 2 + p \quad \text{or} \quad u + 2 = n - p \tag{8.11}$$

According to (8.1) the number of points in the 2D constellation is

$$L = M2^{u+2} = M2^{n-p} \tag{8.12}$$

The number of uncoded bits, u, must be positive, so from (8.11) it follows that

$$u = n - 2 - p \geq 0$$

so

$$0 \leq p \leq n-2 \tag{8.13}$$

and this requires $n \geq 2$. From (8.12) we see that if the constellation size, L, is fixed, increasing p by some integer i requires the number of rings, M, to be increased by the factor 2^i and results in greater implementation complexity.

8.2 Ring Weights and the Number of Frames of Each Weight

Let the rings be labeled from 0 to $M-1$ with ring 0 being closest to the origin and ring $M-1$ furthest from the origin. A frame of N rings will be denoted by $\mathbf{r} = (r_1, r_2, \ldots, r_N)$ with $r_i \in \{0, 1, \ldots, M-1\}$ for $i = 1, \ldots, N$. Each ring must be assigned a positive integer weight $w_1(r_i)$ where the subscript, 1, indicates that the argument is a one-dimensional vector. For example, if the constellation points are scaled to have integer coordinates, the ring weight could be the sum of the squared distances of points in the ring from the origin. Motorola [54] proposed using the weight function $w_1(i) = i$ for $i = 0, \ldots, M-1$. Justification for the Motorola weight function is given in Appendix 8.A. The ring weights should form an increasing sequence, that is,

$$0 \leq w_1(0) < w_1(1) < \ldots < w_1(M-1) \tag{8.14}$$

The weight of a ring frame will be defined as the sum of the component weights, that is,

$$w_N(\mathbf{r}) = \sum_{i=1}^{N} w_1(r_i) \tag{8.15}$$

With the Motorola weight function, the weights of N-tuples can take on all integer values from 0 to $N(M-1)$.

The number of ring frames with a given weight must be known to perform shell mapping. Let $M_p(j)$ be the number of p-tuples of ring indexes, $\mathbf{r} = \{r_1, \ldots, r_p\}$, with total weight $j = w_1(r_1) + \cdots + w_1(r_p)$. The *weight generating function* is defined to be

$$G_p(z) = \sum_k M_p(k) z^k \tag{8.16}$$

EXAMPLE 8.1 Motorola Weight Generating Function

For example, with the Motorola weight function $w_1(i) = i$, the number of 1-tuples of weight j is $M_1(j) = 1$ for $j = 0, \ldots, M-1$ and is 0 otherwise. The corresponding generating function is

$$G_1(z) = \sum_{k=0}^{M-1} z^k = \frac{1 - z^M}{1 - z} \tag{8.17}$$

■

The number of $2p$-tuples of weight j can be computed from the number of p-tuples of each weight in the following way. Each $2p$-tuple can be considered to be the concatenation of a pair of p-tuples . The weight of the $2p$-tuple is the

sum of the weights of the two p-tuples. If the first p-tuple has weight k, then the second p-tuple must have weight $j-k$ to make the total weight equal to j. Thus the number of $2p$-tuples of weight j must be

$$M_{2p}(j) = \sum_{k=0}^{j} M_p(k) M_p(j-k) \tag{8.18}$$

This sum is just a convolution, so the corresponding generating function is

$$G_{2p}(z) = G_p^2(z) \tag{8.19}$$

If the shaping frame length N is a power of 2, $M_N(j)$ can be found by successively applying (8.18) for $2p = 2, 4, \ldots, N$ since the initial sequence $M_1(j)$ is easily determined. The shell mapping encoding and decoding algorithms assume that the intermediate results $M_k(j)$ have been stored in memory for $k = 1, 2, 4, \ldots, 2^n, \ldots, N/2$ and all relevant j.

Another sequence that is required for shell mapping is the number of ring frames with weight less than or equal to j. This will be designated by

$$C_N(j) = \sum_{k=0}^{j} M_N(k) \tag{8.20}$$

These values can also be pre-computed and stored in a table.

The number of ring frames required is 2^K. Thus, the maximum value, J, required for j is the smallest j such that $C_N(j) \geq 2^K$. Then 2^K frames can be selected from the $C_N(J)$ frames. J is called the *shell mapping threshold.*

The maximum possible shell frame weight is $N(M-1)$. However, the shell mapping algorithms only require the values of $C_N(j)$ for $j < J$ which puts a reasonable limit on the storage required. Since the weights of N-dimensional frames cannot exceed J, the weights of shorter subframes must not exceed J also. Therefore, the storage for the values of $M_i(j)$ can be truncated too.

8.3 Lexicographical Ordering of Ring Frames

Binary shaping K-tuples can be assigned to ring frames in many ways. The method we will use is based on a doubling algorithm that successively concatenates pairs of frames into ones of double the length until frames of length N are reached. Therefore, the frame length N must be a power of two. At each iteration, the i-tuples of rings are arranged in an ordered list according to rules which will be stated shortly. The items on the lists are numbered from 0 to $M^i - 1$ and the number assigned to an i-tuple is called its *index.* As a matter of notation, let an i-tuple of scalar ring indexes be

$$\mathbf{r}^{[i]} = [r_1, \ldots, r_k, \ldots, r_i] \text{ with } r_k \in \{0, 1, \ldots, M-1\} \text{ for } k = 1, \ldots, i \tag{8.21}$$

The first and second halves of $\mathbf{r}^{[i]}$ will be designated by

$$\mathbf{r}_1^{[i]} = [r_1, \ldots, r_{i/2}] \text{ and } \mathbf{r}_2^{[i]} = [r_{i/2+1}, \ldots, r_i] \tag{8.22}$$

Let the function $D_i(\mathbf{r}^{[i]})$ be the index of the ring i-tuple $\mathbf{r}^{[i]}$. Notice that $D_i(\mathbf{r}^{[i]})$ is the number of elements on the list below $\mathbf{r}^{[i]}$, that is, with indexes less than $D_i(\mathbf{r}^{[i]})$. The ordered lists are similar to dictionaries with alphabetical ordering and are said to be *lexicographically* ordered.

As a starting point for the doubling procedure, we will arrange the M possible 1-tuples of ring indexes in numerical order. That is, the one-dimensional ring index function is

$$D_1(r) = r \text{ for } r = 0, \ldots, M-1 \tag{8.23}$$

The three rules for ordering ring i-tuples are:

Case 1: Different Weight i-tuples $\mathbf{u}^{[i]}$ and $\mathbf{v}^{[i]}$

List an i-tuple $\mathbf{u}^{[i]}$ below $\mathbf{v}^{[i]}$ if $w_i(\mathbf{u}^{[i]}) < w_i(\mathbf{v}^{[i]})$ where $w_i(\cdot)$ is the i-dimensional weight function.

Case 2: Equal Weight i-tuples but First Halves have Different $i/2$-dimensional Indexes

In this case $w_i(\mathbf{u}^{[i]}) = w_i(\mathbf{v}^{[i]})$ but the first halves of $\mathbf{u}^{[i]}$ and $\mathbf{v}^{[i]}$ are different. List $\mathbf{u}^{[i]}$ below $\mathbf{v}^{[i]}$ if

$$D_{i/2}(\mathbf{u}_1^{[i]}) < D_{i/2}(\mathbf{v}_1^{[i]})$$

Case 3: Equal Weight i-tuples, Indexes of First Halves are the Same

In this case $w_i(\mathbf{u}^{[i]}) = w_i(\mathbf{v}^{[i]})$ and $\mathbf{u}_1^{[i]} = \mathbf{v}_1^{[i]}$. List $\mathbf{u}^{[i]}$ below $\mathbf{v}^{[i]}$ if $D_{i/2}(\mathbf{u}_2^{[i]}) < D_{i/2}(\mathbf{v}_2^{[i]})$.

EXAMPLE 8.2 Shell Frame Ordering for $N = 4$, $M = 3$, Motorola Weight Function

This example illustrates the doubling procedure for ordering ring 4-tuples when there are $M = 3$ rings with indexes 0, 1, and 2. First, the 1-tuples are listed. Then 1-tuples are concatenated to form 2-tuples and ordered. Then the 2-tuples are concatenated to form 4-tuples and ordered.

$D_1(\mathbf{r}^{[1]})$	$\mathbf{r}^{[1]}$	
0	0	$M_1(0) = 1$
1	1	$M_1(1) = 1$
2	2	$M_1(2) = 1$

$D_2(\mathbf{r}^{[2]})$	$\mathbf{r}^{[2]}$		
0	00	$M_2(0) = 1$	
1	01	$M_2(1) = 2$	$C_2(0) = 1$
2	10		
3	02	$M_2(2) = 3$	$C_2(1) = 3$
4	11		
5	20		
6	12	$M_2(3) = 2$	$C_2(2) = 6$
7	21		
8	22	$M2(4) = 1$	$C_2(3) = 8$

$D_4(\mathbf{r})$	r		
0	0000	$M_4(0) = 1$	
1	0001	$M_4(1) = 4$	$C_4(0) = 1$
2	0010		
3	0100		
4	1000		
5	0002	$M_4(2) = 10$	$C_4(1) = 5$
6	0011		
7	0020		
8	0101		
9	0110		
10	1001		
11	1010		
12	0200		
13	1100		
14	2000		
15	0012	$M_4(3) = 16$	$C_4(2) = 15$
16	0021		
17	0102		
18	0111		
19	0120		
20	1002		
21	1011		
22	1020		
23	0201		

24	0210		
25	1101		
26	1110		
27	2001		
28	2010		
29	1200		
30	2100		
31	0022	$M_4(4) = 19$	$C_4(3) = 31$
32	0112		
33	0121		
34	1012		
35	1021		
36	0202		
37	0211		
38	0220		
39	1102		
40	1111		
41	1120		
42	2002		
43	2011		
44	2020		
45	1201		
46	1210		
47	2101		
48	2110		
49	2200		
50	0122	$M_4(5) = 16$	$C_4(4) = 50$
51	1022		
52	0212		
53	0221		
54	1112		
55	1121		
56	2012		
57	2021		
58	1202		
59	1211		
60	1220		
61	2102		
62	2111		
63	2120		

64	2201		
65	2210		
66	0222	$M_4(6) = 10$	$C_4(5) = 66$
67	1122		
68	2022		
69	1212		
70	1221		
71	2112		
72	2121		
73	2202		
74	2211		
75	2220		
76	1222	$M_4(7) = 4$	$C_4(6) = 76$
77	2122		
78	2212		
79	2221		
80	2222	$M_4(8) = 1$	$C_4(7) = 80$
			$C_4(8) = 81$

8.4 The Decoding Algorithm

In principle, encoding and decoding shell frames is quit simple. The encoder uses the block of K shell mapping bits to look up the sequence of N rings in the ordered list while the decoder finds the frame of N rings in the ordered list and outputs its K bit index. In practice, the problem is storing the ordered list. For example, one of the many options in V.34 modems is to use a symbol rate of 2743 Hz and a data rate of 14,400 bits/second with blocks of $K = 30$ shell mapping bits. Each shell frame consists of a sequence of $N = 8$ ring indexes with $M = 14$ or 17 rings. This requires an ordered list with $2^{30} = 1,073,741,824$ entries. We will see in this section and the next that it is not necessary to store the ordered list. Iterative encoding and decoding algorithms will be derived that use stored tables requiring much less memory than the entire ordered list.

In the receiver, the received signal is demodulated and Viterbi decoded to give a maximum likelihood estimate of the transmitted sequence of constellation points. Then frames of N estimated 2D constellation points are quantized into frames of ring indexes.

Given a received frame, $\mathbf{r}$, of N ring indexes, the decoding function $D_N(\mathbf{r})$ computes the index of $\mathbf{r}$ in the list. The decoding function can also be expressed as

$$D_N(\mathbf{r}) = \text{number of frames below } \mathbf{r} \text{ on the list} \tag{8.24}$$

The i-dimensional decoding function can also be expressed in terms of the following function:

$$\begin{aligned} N_i(\mathbf{v}^{[i]}) &= \text{number of } i\text{-tuples } \mathbf{u}^{[i]} \text{ with } w_i(\mathbf{u}^{[i]}) = w_i(\mathbf{v}^{[i]}) \\ &\quad \text{and } \mathbf{u}^{[i]} \text{ below } \mathbf{v}^{[i]} \end{aligned} \tag{8.25}$$

This quantity will be called the offset into the shell of weight $w_i(\mathbf{v}^{[i]})$. Then, the i-dimensional decoding function can be expressed as

$$\begin{aligned} D_i(\mathbf{v}^{[i]}) &= \text{number of } i\text{-tuples below } \mathbf{v}^{[i]} \\ &= \{\text{number of } i\text{-tuples } \mathbf{u}^{[i]} \text{ with } w_i(\mathbf{u}^{[i]}) < w_i(\mathbf{v}^{[i]})\} \\ &\quad + \{\text{number of } i\text{-tuples } \mathbf{u}^{[i]} \text{ with } w_i(\mathbf{u}^{[i]}) = w_i(\mathbf{v}^{[i]}) \\ &\quad \text{and } \mathbf{u}^{[i]} \text{ below } \mathbf{v}^{[i]}\} \\ &= C_i[w_i(\mathbf{v}_i^{[i]}) - 1] + N_i(\mathbf{v}^{[i]}) \end{aligned} \tag{8.26}$$

It will become clear that $D_i(\cdot)$ is not needed until the final step when $i = N$. Also, $C_i(\cdot)$ is needed only for $i = N$. Finally, we will see in the next paragraph how to recursively compute $N_i(\cdot)$ for $i = 2, 4, \ldots, N$ using $N_{i/2}(\cdot)$ and $M_{i/2}(\cdot)$. Then $D_N(\cdot)$ can be computed by (8.26) for $i = N$.

The vectors counted in $N_i(\mathbf{v}^{[i]})$ can be partitioned into the following three types:

Type 1: First Halves Differ in Weight

Given a vector $\mathbf{v}^{[i]}$, consider the set of ring i-tuples, $\mathbf{u}^{[i]}$,

$$\left\{\mathbf{u}^{[i]} \,|\, w_i(\mathbf{u}^{[i]}) = w_i(\mathbf{v}^{[i]}), \text{ and } w_{i/2}(\mathbf{u}_1^{[i]}) < w_{i/2}(\mathbf{v}_1^{[i]})\right\} \tag{8.27}$$

The number of elements in this set is

$$a_1(\mathbf{v}^{[i]}) = \sum_{k=0}^{w_{i/2}(\mathbf{v}_1^{[i]})-1} M_{i/2}(k) M_{i/2}[w_i(\mathbf{v}^{[i]}) - k] \tag{8.28}$$

with the understanding that the sum is zero if the upper limit is negative.

Type 2: First Halves Differ but Have Same Weight

Consider the set of ring i-tuples

$$\left\{\mathbf{u}^{[i]} \,|\, w_i(\mathbf{u}^{[i]}) = w_i(\mathbf{v}^{[i]}), \text{ and } w_{i/2}(\mathbf{u}_1^{[i]}) = w_{i/2}(\mathbf{v}_1^{[i]}), \text{ and } D_{i/2}(\mathbf{u}_1^{[i]}) < D_{i/2}(\mathbf{v}_1^{[i]})\right\} \tag{8.29}$$

According to (8.25), the number of choices for $\mathbf{u}_1^{[i]}$ is $N_{i/2}(\mathbf{v}_1^{[i]})$. The total weight must be $w_i(\mathbf{v}^{[i]})$, so the number of choices for $\mathbf{u}_2^{[i]}$ is

$$M_{i/2}[w_i(\mathbf{v}^{[i]}) - w_{i/2}(\mathbf{v}_1^{[i]})] \tag{8.30}$$

Thus, the number of type 2 vectors is

$$a_2(\mathbf{v}^{[i]}) = N_{i/2}(\mathbf{v}_1^{[i]}) M_{i/2}[w_i(\mathbf{v}^{[i]}) - w_{i/2}(\mathbf{v}_1^{[i]})] \tag{8.31}$$

Notice also that the weight of $\mathbf{v}_2^{[i]}$ is

$$w_{i/2}(\mathbf{v}_2^{[i]}) = w_i(\mathbf{v}^{[i]}) - w_{i/2}(\mathbf{v}_1^{[i]}) \tag{8.32}$$

so

$$a_2(\mathbf{v}^{[i]}) = N_{i/2}(\mathbf{v}_1^{[i]}) M_{i/2}[w_{i/2}(\mathbf{v}_2^{[i]})] \tag{8.33}$$

Type 3: First Halves are Identical

The number of vectors in the set of ring i-tuples

$$\left\{ \mathbf{u}^{[i]} \,|\, w_i(\mathbf{u}^{[i]}) = w_i(\mathbf{v}^{[i]}), \text{ and } \mathbf{u}_1^{[i]} = \mathbf{v}_1^{[i]}, \right.$$
$$\left. \text{and } D_{i/2}(\mathbf{u}_2^{[i]}) < D_{i/2}(\mathbf{v}_2^{[i]}) \right\} \tag{8.34}$$

is

$$a_3(\mathbf{v}^{[i]}) = N_{i/2}(\mathbf{v}_2^{[i]}) \tag{8.35}$$

Adding the formulas for the three cases gives

$$N_i(\mathbf{v}^{[i]}) = \sum_{k=0}^{w_{i/2}(\mathbf{v}_1^{[i]})-1} M_{i/2}(k) M_{i/2}[w_i(\mathbf{v}^{[i]}) - k]$$
$$+ N_{i/2}(\mathbf{v}_1^{[i]}) M_{i/2}[w_{i/2}(\mathbf{v}_2^{[i]})] + N_{i/2}(\mathbf{v}_2^{[i]}) \tag{8.36}$$

with the understanding that the summation is zero when the upper limit is negative, that is, when $w_{i/2}(\mathbf{v}_1^{[i]}) = 0$.

The decoding function can be performed iteratively. First, $\mathbf{r} = \mathbf{v}^{[N]}$ is divided into $N/2$ successive pairs of ring indexes and $N_2(\cdot)$ is computed for each of the pairs using (8.36). According to (8.14), the weights assigned to the ring indexes must form an increasing sequence, so there is just one ring of each weight and $N_1(\mathbf{v}_1^{[2]}) = N_1(\mathbf{v}_2^{[2]}) = 0$. In this case (8.36) reduces to

$$N_2(\mathbf{v}^{[2]}) = \sum_{k=0}^{w_1(\mathbf{v}_1^{[2]})-1} M_1[w_2(\mathbf{v}^{[2]}) - k] \tag{8.37}$$

Changing the summation index from k to $k' = w_2(\mathbf{v}^{[2]}) - k$ gives the alternative form

$$N_2(\mathbf{v}^{[2]}) = \sum_{k=w_2(\mathbf{v}^{[2]})-w_1(\mathbf{v}_1^{[2]})+1}^{w_2(\mathbf{v}^{[2]})} M_1(k) = \sum_{k=w_1(\mathbf{v}_2^{[2]})+1}^{w_1(\mathbf{v}_1^{[2]})+w_1(\mathbf{v}_2^{[2]})} M_1(k) \tag{8.38}$$

with the understanding that the summation is zero when the upper limit is less than the lower limit.

Next adjacent 2-tuples are concatenated to form $N/4$ 4-tuples and $N_4(\cdot)$ is computed for each. The doubling procedure is repeated until $N_N(\mathbf{r})$ is computed. Then the decoding function $D_N(\mathbf{r})$ is computed by (8.26) with $i = N$. In implementing this procedure, we assume that the $M_{i/2}(\cdot)$ and $C_N(\cdot)$ functions have been pre-computed and stored in tables.

EXAMPLE 8.3 N=2 and the Motorola Weight Function

Let the one-dimensional weight function be

$$w_1(r) = r \text{ for } r = 0, \ldots, M-1 \tag{8.39}$$

and designate 2-tuples by $\mathbf{v}^{[2]} = [r_1, r_2]$. The number of 1-tuples of each weight is

$$M_1(k) = \begin{cases} 1 \text{ for } k = 0, \ldots, M-1 \\ 0 \text{ elsewhere} \end{cases} \tag{8.40}$$

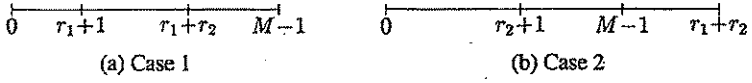
For this example,

$$\mathbf{v}_1^{[2]} = r_1 \text{ and } \mathbf{v}_2^{[2]} = r_2 \tag{8.41}$$

Then according to (8.38), the shell offset is

$$N_2(\mathbf{v}^{[2]}) = \begin{cases} \sum_{k=r_2+1}^{r_1+r_2} M_1(k) & \text{for } r_1 > 0 \\ 0 & \text{for } r_1 = 0 \end{cases} \tag{8.42}$$

This sum can be computed with the aid of Figure 8.2. For Case 1 where



(a) Case 1 (b) Case 2

Figure 8.2. Limits for Computing the Summation

$r_1 + r_2 \le M - 1$, the result is

$$N_2(\mathbf{v}^{[2]}) = \sum_{k=r_2+1}^{r_1+r_2} 1 = r_1 \tag{8.43}$$

For Case 2 where $r_2 + 1 \le M - 1$ and $r_1 + r_2 > M - 1$, the result is

$$N_2(\mathbf{v}^{[2]}) = \sum_{k=r_2+1}^{M-1} 1 = M - 1 - r_2 \tag{8.44}$$

Putting the results together gives

$$N_2([r_1, r_2]) = \begin{cases} r_1 & \text{for } 0 \le r_1 + r_2 \le M - 1 \\ M - 1 - r_2 & \text{for } M \le r_1 + r_2 \le r_1 + M - 2 \\ 0 & \text{otherwise} \end{cases} \tag{8.45}$$

As specific examples we will apply (8.45) to the 2-tuples $[2, 1]$ and $[1, 1]$ when there are $M = 3$ rings. According to this formula,

$$N_2([2, 1]) = M - 1 - r_2 = 1 \text{ and } N_2([1, 1]) = r_1 = 1 \tag{8.46}$$

This agrees with the section of the tables in Example 8.2 for 2-tuples. ∎

EXAMPLE 8.4 Decoding Example for $N = 4$, $M = 3$ with the Motorola Weight Function

As an example of decoding 4-tuples, consider the frame $\mathbf{v}^{[4]} = [2, 1, 1, 1]$. The two 2-tuples required are $\mathbf{v}_1^{[4]} = [2, 1]$ and $\mathbf{v}_2^{[4]} = [1, 1]$. The weight of this frame is $w_4(\mathbf{v}^{[4]}) = 5$. According to (8.36), the tables for Example 8.2, and the results of Example 8.3 for $N = 2$, the offset must be

$$\begin{aligned} N_4(\mathbf{v}^{[4]}) &= \sum_{k=0}^{w_2(\mathbf{v}_1^{[4]})-1} M_2(k) M_2[w_4(\mathbf{v}^{[4]}) - k] + N_2(\mathbf{v}_1^{[4]}) M_2[w_2(\mathbf{v}_2^{[4]})] \\ &\quad + N_2(\mathbf{v}_2^{[4]}) \\ &= \sum_{k=0}^{3-1} M_2(k) M_2(5 - k) + N_2([2, 1]) M_2(2) + N_2([1, 1]) \\ &= (1 \times 0 + 2 \times 1 + 3 \times 2) + 1 \times 3 + 1 = 12 \end{aligned} \tag{8.47}$$

According to the table on page 165, the number of 4-tuples with weight less than 5 is $C_4(4) = 50$. Thus the list index of $\mathbf{v}^{[4]}$ is $D_4(\mathbf{v}^{[4]}) = C_4(4) + N_4(\mathbf{v}^{[4]}) = 62$. As it should be, this is the same value as in the list on page 165. ∎

8.5 The Encoding Algorithm

The encoding algorithm maps binary K-tuples of shaping bits into N-tuples of ring indexes. The mapping is based on the ordering of ring frames described in Section 8.3. Let the shaping K-tuple be the binary representation for the list index $D_N(\mathbf{v}^{[N]})$. The problem is to find the shell frame $\mathbf{v}^{[N]}$ by an algorithm of reasonable complexity so a huge ordered list does not have to be stored in memory. According to (8.26)

$$D_N(\mathbf{v}^{[N]}) = C_N[w_N(\mathbf{v}^{[N]}) - 1] + N_N(\mathbf{v}^{[N]}) \tag{8.48}$$

Remember that $N_N(\mathbf{v}^{[N]})$ is the number of frames with the same weight as $\mathbf{v}^{[N]}$ that are below it on the list. Also, $M_N(w_N(\mathbf{v}^{[n]}))$ is the number of frames with the weight of $\mathbf{v}^{[N]}$. Therefore,

$$0 \le N_N(\mathbf{v}^{[N]}) < M_N[w_N(\mathbf{v}^{[N]})] \tag{8.49}$$

Also,

$$C_N[w_N(\mathbf{v}^{[N]})] = C_N[w_N(\mathbf{v}^{[N]}) - 1] + M_N[w_N(\mathbf{v}^{[N]})] \tag{8.50}$$

Thus,

$$C_N[w_N(\mathbf{v}^{[N]}) - 1] \le D_N(\mathbf{v}^{[N]}) < C_N[w_N(\mathbf{v}^{[N]})] \tag{8.51}$$

Therefore, the index D_N always falls in an interval bracketed by a pair of successive C_N's.

A sequence of steps for encoding will now be presented. The problem is to find the frame of N ring indexes $\mathbf{v}^{[N]}$ given a block of K input bits with the value $D_N(\mathbf{v}^{[N]})$.

Encoding Step 1: Find the weight of $\mathbf{v}^{[N]}$

Let $\Omega_N = \{\omega_{N,n}; n = 1, \ldots, \nu_1\}$ be the set of distinct weights found in the set of 2^K possible ring frames. Assume these weights are arranged in increasing order, that is, $\omega_{N,i} < \omega_{N,j}$ for $i < j$. Based on (8.51), the weight of $\mathbf{v}^{[N]}$ must be

$$w_N(\mathbf{v}^{[N]}) = \max\{\omega_{N,n} \,|\, C_N(\omega_{N,n-1}) \le D_N(\mathbf{v}^{[N]})\} \tag{8.52}$$

The search for the maximum can be implemented in a computer program by starting with the minimum weight ω_1 and testing that the inequality is satisfied for each successively larger weight until it is violated for the first time at $n = n_1 + 1$. Then $w_N(\mathbf{v}^{[N]}) = w_{n_1}$.

Encoding Step 2: Compute the Offset

Once the weight is known, the offset can be computed as

$$N_N(\mathbf{v}^{[N]}) = D_N(\mathbf{v}^{[N]}) - C_N[w_N(\mathbf{v}^{[N]}) - 1] \tag{8.53}$$

Encoding Step 3: Finding the Weights of $\mathbf{v}_1^{[N]}$ and $\mathbf{v}_2^{[N]}$

According to (8.36)

$$\begin{aligned} N_N(\mathbf{v}^{[N]}) &= \sum_{k=0}^{w_{N/2}(\mathbf{v}_1^{[N]})-1} M_{N/2}(k) M_{N/2}[w_N(\mathbf{v}^{[N]}) - k] \\ &\quad + N_{N/2}(\mathbf{v}_1^{[N]}) M_{N/2}[w_{N/2}(\mathbf{v}_2^{[N]})] \\ &\quad + N_{N/2}(\mathbf{v}_2^{[N]}) \end{aligned} \tag{8.54}$$

Of the weight $w_N(\mathbf{v}^{[N]})$ N-tuples $\mathbf{u}^{[N]}$, the number with the same weight as $\mathbf{v}^{[N]}$ in the first half, that is, $w_{N/2}(\mathbf{u}_1^{[N]}) = w_{N/2}(\mathbf{v}_1^{[N]})$, is

$$M_{N/2}[w_{N/2}(\mathbf{v}_1^{[N]})] M_{N/2}[w_{N/2}(\mathbf{v}_2^{[N]})] \tag{8.55}$$

where $w_{N/2}(\mathbf{u}_2^{[N]}) = w_{N/2}(\mathbf{v}_2^{[N]}) = w_N(\mathbf{v}^{[N]}) - w_{N/2}(\mathbf{v}_1^{[N]})$. The sum of the second and third terms on the right hand side of (8.54) is the number of N-tuples below $\mathbf{v}^{[N]}$ with the same total weight as $\mathbf{v}^{[N]}$ and the same weight in the first half as $\mathbf{v}_1^{[N]}$. This sum is bounded by

$$\begin{aligned} 0 \le N_{N/2}(\mathbf{v}_1^{[N]}) M_{N/2}[w_{N/2}(\mathbf{v}_2^{[N]})] + N_{N/2}(\mathbf{v}_2^{[N]}) \\ < M_{N/2}[w_{N/2}(\mathbf{v}_1^{[N]})] M_{N/2}[w_{N/2}(\mathbf{v}_2^{[N]})] \end{aligned} \tag{8.56}$$

According to (8.54)

$$\begin{aligned} &N_{N/2}(\mathbf{v}_1^{[N]}) M_{N/2}[w_{N/2}(\mathbf{v}_2^{[N]})] + N_{N/2}(\mathbf{v}_2^{[N]}) \\ &= N_N(\mathbf{v}^{[N]}) - \sum_{k=0}^{w_{N/2}(\mathbf{v}_1^{[N]})-1} M_{N/2}(k) M_{N/2}[w_N(\mathbf{v}^{[N]}) - k] \end{aligned} \tag{8.57}$$

Therefore, (8.56) is equivalent to

$$\begin{aligned} 0 \le N_N(\mathbf{v}^{[N]}) - \sum_{k=0}^{w_{N/2}(\mathbf{v}_1^{[N]})-1} M_{N/2}(k) M_{N/2}[w_N(\mathbf{v}^{[N]}) - k] \\ < M_{N/2}[w_{N/2}(\mathbf{v}_1^{[N]})] M_{N/2}[w_{N/2}(\mathbf{v}_2^{[N]})] \end{aligned} \tag{8.58}$$

Notice that the term on the far right is just the additional term that would be added to the summation over k if the upper limit were extended from $k = w_{N/2}(\mathbf{v}_1^{[N]}) - 1$ to $w_{N/2}(\mathbf{v}_1^{[N]})$. Therefore, the last inequality can be written as

$$\sum_{k=0}^{w_{N/2}(\mathbf{v}_1^{[N]})-1} M_{N/2}(k) M_{N/2}[w_N(\mathbf{v}^{[N]}) - k] \le N_N(\mathbf{v}^{[N]})$$

$$< \sum_{k=0}^{w_{N/2}(\mathbf{v}_1^{[N]})} M_{N/2}(k) M_{N/2}[w_N(\mathbf{v}^{[N]}) - k] \tag{8.59}$$

Thus, the offset $N_N(\mathbf{v}^{[N]})$ is bracketed by the sum for two successive values of the upper limit on k.

Let $\Omega_{N/2} = \{\omega_{N/2,n}; n = 1, \ldots, \nu_2\}$ be the distinct set of weights for the $N/2$-tuples of ring indexes. Again, assume the weights are arranged in increasing order. Suppose $w_{N/2}(\mathbf{v}_1^{[N]}) = \omega_{N/2,\mu}$. Then (8.59) can be written as

$$\begin{aligned} \sum_{k=0}^{\mu-1} M_{N/2}(\omega_{N/2,k}) M_{N/2}[w_N(\mathbf{v}^{[N]}) - \omega_{N/2,k}] \le N_N(\mathbf{v}^{[N]}) \\ < \sum_{k=0}^{\mu} M_{N/2}(\omega_{N/2,k}) M_{N/2}[w_N(\mathbf{v}^{[N]}) - \omega_{N/2,k}] \end{aligned} \tag{8.60}$$

Therefore, the weight of $\mathbf{v}_1^{[N]}$ must be

$$\begin{aligned} w_{N/2}(\mathbf{v}_1^{[N]}) = \max \Bigg\{ \omega_{N/2,n} \,|\, \sum_{k=0}^{n-1} M_{N/2}(\omega_{N/2,k}) M_{N/2}[w_N(\mathbf{v}^{[N]}) \\ - \omega_{N/2,k}] \le N_N(\mathbf{v}^{[N]}) \Bigg\} \end{aligned} \tag{8.61}$$

The weight of the second half of $\mathbf{v}^{[N]}$ can then be computed as

$$w_{N/2}(\mathbf{v}_2^{[N]}) = w_N(\mathbf{v}^{[N]}) - w_{N/2}(\mathbf{v}_1^{[N]}) \tag{8.62}$$

Encoding Step 4: Compute the Partial Offset

Now the following partial offset d can be computed:

$$\begin{aligned} d &= N_{N/2}(\mathbf{v}_1^{[N]}) M_{N/2}[w_{N/2}(\mathbf{v}_2^{[N]})] + N_{N/2}(\mathbf{v}_2^{[N]}) \\ &= N_N(\mathbf{v}^{[N]}) - \sum_{k=0}^{w_{N/2}(\mathbf{v}_1^{[N]})-1} M_{N/2}(k) M_{N/2}[w_N(\mathbf{v}^{[N]}) - k] \end{aligned} \tag{8.63}$$

Note that this is the same as (8.57).

Encoding Step 5: Compute the Offsets of the First and Second Halves

Remember that $N_{N/2}(\mathbf{v}_2^{[n]})$ is the number of weight $w_{N/2}(\mathbf{v}_2^{[N]})$ $N/2$-tuples below $\mathbf{v}_2^{[N]}$. The total number of $N/2$-tuples with weight $w_{N/2}(\mathbf{v}_2^{[N]})$ is

$$M_{N/2}[w_{N/2}(\mathbf{v}_2^{[N]})]$$

so

$$0 \le N_{N/2}(\mathbf{v}_2^{[n]}) < M_{N/2}[w_{N/2}(\mathbf{v}_2^{[N]})] \tag{8.64}$$

Similarly

$$0 \le N_{N/2}(\mathbf{v}_1^{[n]}) < M_{N/2}[w_{N/2}(\mathbf{v}_1^{[N]})] \tag{8.65}$$

Dividing d by $M_{N/2}[w_{N/2}(\mathbf{v}_2^{[N]})]$, we see that $N_{N/2}(\mathbf{v}_2^{[n]})$ is the remainder and $N_{N/2}(\mathbf{v}_1^{[n]})$ is the quotient. Thus, to complete Step 5, compute the quotient

$$N_{N/2}(\mathbf{v}_1^{[n]}) = \left\lfloor \frac{d}{M_{N/2}[w_{N/2}(\mathbf{v}_2^{[N]})]} \right\rfloor \tag{8.66}$$

where $\lfloor z \rfloor$ is the integer part of z, and the remainder

$$N_{N/2}(\mathbf{v}_2^{[n]}) = d - N_{N/2}(\mathbf{v}_1^{[N]}) M_{N/2}[w_{N/2}(\mathbf{v}_2^{[N]})] \tag{8.67}$$

Encoding Step 6: Iterate the Procedure

Steps 3, 4, and 5 can now be applied to the two $N/2$-tuples to find the weights and offsets of four $N/4$-tuples, and the procedure can be repeated until 1-tuples are reached. Then the ring index for a 1-tuple will be obvious from its weight $w_1(\cdot)$ since there is only one 1-tuple of each weight.

EXAMPLE 8.5 Encoding 2-tuples with the Motorola Weight Function

For the Motorola weight function, it is not necessary to decompose 2-tuples into 1-tuples since the results have been analytically computed and can be found from (8.45). Let $\mathbf{v} = [r_1, r_2]$ so $w_2(\mathbf{v}) = r_1 + r_2$. Rearranging (8.45) to find r_1 and r_2 gives

$$r_1 = \begin{cases} N_2(\mathbf{v}) & \text{for } w_2(\mathbf{v}) \le M-1 \\ w_2(\mathbf{v}) + N_2(\mathbf{v}) - (M-1) & \text{for } M-1 < w_2(\mathbf{v}) \end{cases} \tag{8.68}$$

$$r_2 = w_2(\mathbf{v}) - r_1 \tag{8.69}$$

To be more specific, let us find the 2-tuple $\mathbf{v}$ corresponding to the list index $D_2(\mathbf{v}) = 7$ when $M = 3$ rings are used. In Step 1 we find from the table on page 164 that $C_2(2) = 6$ and $C_2(3) = 8$ so $w_2(\mathbf{v}) = 3$. Finding the offset using the formula in Step 2 gives $N_2(\mathbf{v}) = D_2(\mathbf{v}) - C_2(2) = 1$. In this case, $w_2(\mathbf{v}) > M-1$, so according to (8.68)

$$\begin{aligned} r_1 &= w_2(\mathbf{v}) + N_2(\mathbf{v}) - (M-1) = 3 + 1 - 2 = 2 \\ r_2 &= w_2(\mathbf{v}) - r_1 = 3 - 2 = 1 \end{aligned}$$

If instead of using the direct formula for 2-tuples, Step 3 is performed, we find from (8.61) that $w_1(r_1) = 2$ so $r_1 = 2$ and $r_2 = w_2(\mathbf{v}) - w_1(r_1) = 1$ as before.

The encoded frame $\mathbf{v} = [2, 1]$ agrees with the table on page 164. ∎

EXAMPLE 8.6 Encoding for $N = 4$, $M = 3$, Motorola Weight Function

Suppose the K shell mapping bits are equivalent to the decimal list index $D_4(\mathbf{v}^{[4]}) = 13$. The encoding procedure is:

Step 1

From the tables on page 164 we find that

$$C_4(1) = 5 \le 13 < C_4(2) = 15$$

According to the rules for Step 1, $w_4(\mathbf{v}^{[4]}) = 2$.

Step 2

The offset is $N_4(\mathbf{v}^{[4]}) = D_4(\mathbf{v}^{[4]}) - C_4(1) = 13 - 5 = 8$.

Step 3 Finding Weights of First and Second Halves

$$\begin{aligned} M_2(0)M_2(2) \;+\; & M_2(1)M_2(1) = 1 \times 3 + 2 \times 2 = 7 \le 8 \\ < \; & M_2(0)M_2(2) + M_2(1)M_2(1) + M_2(2)M_2(0) = 10 \end{aligned}$$

Thus $w_2(\mathbf{v}_1^{[4]}) = 2$ and $w_2(\mathbf{v}_2^{[4]}) = w_4(\mathbf{v}^{[r]}) - w_2(\mathbf{v}_1^{[4]}) = 2 - 2 = 0$.

Step 4

The partial offset is $d = 8 - 7 = 1$.

Step 5 Find Offsets of First and Second Halves

$$M_2(w_2(\mathbf{v}_2^{[4]})) = M_2(0) = 1$$

$$N_2(\mathbf{v}_1^{[4]}) = \lfloor d / M_2(w_2(\mathbf{v}_2^{[4]})) \rfloor = 1$$

$$N_2(\mathbf{v}_2^{[4]}) = d - M_2(w_2(\mathbf{v}_2^{[4]})) N_2(\mathbf{v}_1^{[4]}) = 0$$

Applying (8.68) and(8.69) to both halves, we find that

$$r_1 = N_2(\mathbf{v}_1^{[4]}) = 1, \quad r_2 = w_2(\mathbf{v}_1^{[4]}) - r_1 = 2 - 1 = 1$$

$$r_3 = N_2(\mathbf{v}_2^{[4]}) = 0, \quad r_4 = w_2(\mathbf{v}_2^{[4]}) - r_3 = 0 - 0 = 0$$

Therefore, the encoded ring frame is $\mathbf{v}^{[4]} = [1100]$ which is the result expected from the table on page 164. ∎

EXAMPLE 8.7 Ring Frequency of Occurrence for $K = 18$, $N = 8$, $M = 7$, and the Motorola Weight Function

This example illustrates how shell mapping causes the inner rings to occur with higher probability than the outer rings. A shell frame size of $N = 8$ was chosen because it is the size used by V.34 modems. The number of shell mapping bits was $K = 18$, so 2^{18} distinct shell frames are required. The number of rings was selected to be $M = 7$. According to (8.5) the constellation expansion ratio is CER $= M2^{-K/N} = 1.472$. All of the 2^{18} ring frames were generated by the FORTRAN program listed in Appendix 8.B and the number of times each ring occurred in each baud position was counted by the program. The numerical results are shown in Table 8.7. The values in each column differ because all the ring frames with largest weight are not used. The largest weight was found to be $J = 15$, so the tables required for shell mapping can be truncated at $J - 1 = 14$.

The results for baud position 1 are plotted in Figure 8.1. Positions 2 through 8 have similar plots. The ring frequencies of occurrence appear to decrease exponentially from the inner to outer ring. In fact, it can be shown that if the rings are concentric annuli of equal area and the constellation points have a two-dimensional circular Gaussian probability density function, the ring probabilities actually do decrease exponentially.

Table 8.1. Number of Times Rings Occur for $K = 18$ and $M = 7$

	Baud Position							
Ring	1	2	3	4	5	6	7	8
0	91502	91501	91482	91478	89117	89043	88445	87997
1	64608	64607	64595	64595	63621	63695	63845	63845
2	43871	43870	43865	43865	43941	43941	44165	44613
3	28522	28522	28524	28524	29290	29290	29514	29514
4	17653	17654	17662	17662	18608	18608	18608	18608
5	10325	10326	10337	10341	11188	11188	11188	11188
6	5663	5664	5679	5679	6379	6379	6379	6379

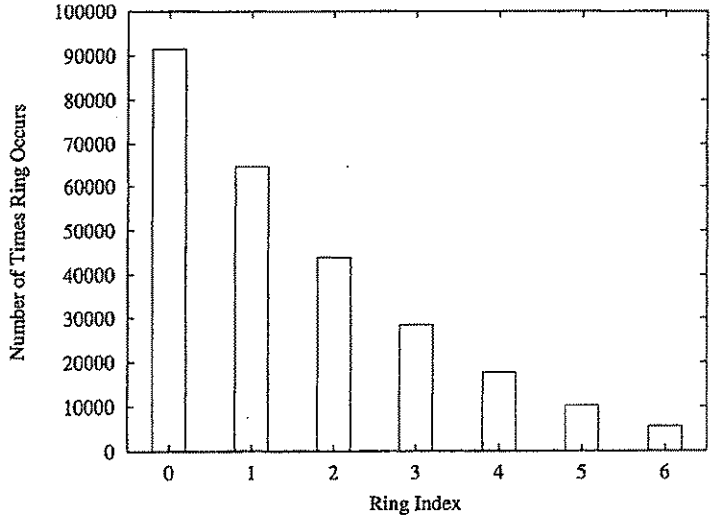


Figure 8.3. Ring Frequency in Baud 1 for $k = 2, p = 2, K = 18, M = 7, N = 8$

8.A Justification for the Motorola Weight Function

Suppose that the rings are formed from M concentric circles as shown in Figure 8.4. The M concentric circles form shaping rings labeled $0, 1, \ldots, M-1$. The radii $R_1, \ldots, R_M$ are selected so the rings all have the same area. Let the largest radius be $R_M = R$. Then, the area of ring 0 must be

$$A_0 = \pi R_1^2 = \pi R^2/M \tag{8.70}$$

so

$$R_1^2 = R^2/M \tag{8.71}$$

The area enclosed by the outer circle of ring k is

$$A_k = \pi R_{k+1}^2 = (k+1)A_0 = (k+1)\pi R_1^2 \tag{8.72}$$

so

$$R_{k+1}^2 = (k+1)R_1^2 = (k+1)R^2/M \tag{8.73}$$

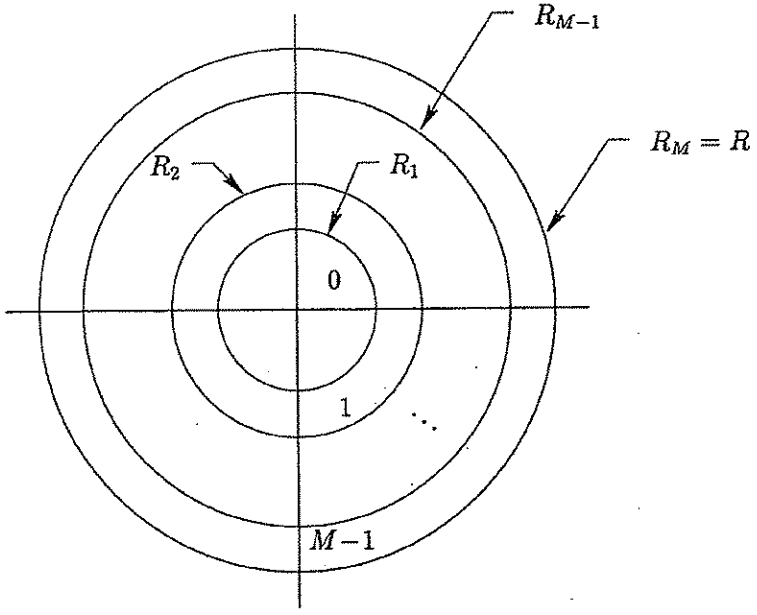


Figure 8.4. Concentric Shaping Rings

The average power for ring k, assuming points are uniformly distributed over the ring, is

$$\begin{aligned} P_k &= \int_{R_k}^{R_{k+1}} r^2 \frac{dA}{A_k} = \int_{R_k}^{R_{k+1}} r^2 \frac{2\pi r\, dr}{\pi(R_{k+1}^2 - R_k^2)} = \frac{R_{k+1}^4 - R_k^4}{2(R_{k+1}^2 - R_k^2)} \\ &= \frac{1}{2}(R_{k+1}^2 + R_k^2) \end{aligned} \tag{8.74}$$

Using (8.73) this reduces to

$$P_k = \frac{R^2}{M}(k + 0.5) \tag{8.75}$$

The average power in a ring frame $\mathbf{v} = [r_1, r_2, \ldots, r_N]$ is

$$P(\mathbf{v}) = \sum_{k=1}^{N} P_{r_k} = \frac{R^2}{M}\left(0.5N + \sum_{k=1}^{N} r_k\right) \tag{8.76}$$

Thus, any ring sequence with the same sum has the same average power. Also, the average power increases monotonically with the sum. This justifies using the ring indexes as a weight function.

8.B Shell Mapping Program

An example FORTRAN program for $N = 8$ shell mapping is included in this appendix. The program allows a constellation expansion ratio of up to 1.5. However, the V.34 Recommendation only allow a CER up to 1.25. An equivalent C program is included in the Motorola CCITT contribution [54]. The variables G2, G4, and G8 in this program correspond to $M_2(\cdot)$, $M_4(\cdot)$, and $M_8(\cdot)$ in this chapter. The variable Z8(I) is the same as $C_8(i-1)$ and the offset R8 corresponds to $N_8(\cdot)$.

```
C   PROGRAM SHELLMAP.FOR
C     THIS PROGRAM PERFORMS SHELL MAPPING ENCODING AND DECODING
C     AS DESCRIBED IN THE CCITT PAPER
C     MOTOROLA INFORMATION SYSTEMS, "SIGNAL MAPPING AND SHAPING
C     FOR V.FAST,"  QUESTION 3/XVII, WP XVII/1, MAY 1992
C
C     WRITTEN BY S.A. TRETTER
C
      INTEGER*4 P,M,M1,I0,G2(0:200),G4(0:200),Z8(0:200)
      INTEGER*4 G8(0:200),X,ZMAX,R8,R4,R41,R42,S0,S,T
      INTEGER*4 R2I1,R2I2,S0MT0,R2I,XOUT,NCOUNT(0:24,8)
      Write(*,'(A\)') ' ENTER  k from {0,1,...,7}, p from {0,1,2,3}: '
      Read(*,*) KLC,p
      K=8*p+KLC
      WRITE(*,1)  K
1     FORMAT(1H ,'  The number of shaping bits is K = ',I4)
      FMLOW = 2.**(K/8.)
      FMHIGH = 1.5*FMLOW
      WRITE(*,'('' Enter an integer M in the range ['',F6.2,'', '',
     1  F6.2,'' ] :  '',\)') FMLOW,FMHIGH
      READ(*,*) M
C
C  GENERATE G2, G4, Z8 TABLES
C
      DO 10 I=0,M-1
        G2(I) = I+1
10      G2(2*M-2-I)=I+1
C
      DO 20 I=0,2*(M-1)
        G4(I)=0
        DO 30 KK=0,I
30        G4(I) = G4(I)+G2(KK)*G2(I-KK)
        G4(4*(M-1)-I) = G4(I)
20    CONTINUE
C
      DO 40 I=0,4*(M-1)
        G8(I)=0
        DO 50 KK=0,I
50        G8(I) = G8(I) + G4(KK)*G4(I-KK)
        G8(8*(M-1)-I) = G8(I)
40    CONTINUE
C
      ZMAX = 2**K
```



```
      Z8(0) = 0
      DO 60 I=1,8*(M-1)+1
        Z8(I) = Z8(I-1) + G8(I-1)
        IF(Z8(I).GT.ZMAX) THEN
          IMAX = I-1
          GOTO 70
        ELSEIF(Z8(I).EQ.ZMAX) THEN
          IMAX = I
          GO TO 70
        ENDIF
60    CONTINUE
C
70    CONTINUE
      WRITE(*,*) ' '
      WRITE(*,*) '    I    G2(I)     G4(I)       G8(I)       Z8(I)'
      DO 80 I=0,IMAX
80      WRITE(*,81) I,G2(I),G4(I),G8(I),Z8(I)
81    FORMAT(1H ,I5,4X,Z4,4X,Z8,4X,Z8,4X,Z8)
C*********************************************************************
C
C  ENCODE THE K SHAPING BITS INTO A RING SEQUENCE
C       {m1,m2,m3,m4,m5,m6,m7,m8}
C
      WRITE(*,*) ' '
      WRITE(*,*) '     2**K = ' ,2**K
      WRITE(*,*) ' '
C       WRITE(*,90) 2**K -1
C90     FORMAT(' ENTER THE K SHAPING BITS REPRESENTED BY AN INTEGER'
C     + /'  IN THE RANGE [0, 2**K - 1 = ',I10,' ]: '\)
C       READ(*,*) X
C=====================================================================
      DO 9999 IX = 0,2**K-1
       X = IX
C=====================================================================
C.....................................................................
C     FIRST FIND I0 WHICH IS THE MAXIMUM I SUCH THAT Z8(I).LE.X .
C     I0 IS THE WEIGHT OF THE 8-TUPLE RING SEQUENCE.
C
      DO 100 I=0,IMAX
        IF(Z8(I).LE.X) THEN
          I0=I
        ELSE
          GO TO 105
        ENDIF
100   CONTINUE
C
105   CONTINUE
C.....................................................................
C     COMPUTE R8 WHICH IS THE POSITION OF X IN THE LIST OF
C     8-TUPLES OF WEIGHT I0
C
      R8 = X - Z8(I0)
C
C     NOTICE THAT  Z8(I0) .LE. X .LE. Z8(I0+1)-1  SO
C      0 .LE. X - Z8(I0) .LE. Z8(I0+1) - Z8(I) -1
C     OR
C      0 .LE. R8 .LE. G8(I0) - 1
```

```
C.....
C   NOW CONSIDER (m1,m2,m3,m4,m5,m6,m7,m8} TO BE THE CONCATENATION
C   OF THE TWO 4-TUPLES {m1,m2,m3,m4} and {m5,m6,m7,m8}.  THEN, THE
C   NUMBER OF 8-TUPLES OF WEIGHT I IS
C
C      G8(I) = SUM FROM S=0 TO I [G4(S)*G4(I-S)]
C
C   LET W41 = WEIGHT{m1,m2,m3,m4} and W42 = WEIGHT{m5,m6,m7,m8}.
C   THEN THE 8-TUPLES OF WEIGHT I0 CAN BE PARTITIONED INTO I0+1
C   SUBSETS CONSISTING OF THE CONCATENATION OF PAIRS OF 4-TUPLES
C   WITH W41=S AND W42=I0-S FOR S=0,...I0.  SUBSET S CONTAINS
C   G4(S)*G4(I0-S) PAIRS.  IF THE PAIRS ARE LABELLED SEQUENTIALLY,
C   THEN R8 MUST BE A LABEL IN THE SUBSET S0 WHICH IS THE LARGEST
C   INTEGER SUCH THAT
C      SUM FROM {S=0 TO S0-1} [G4(S)*G4(I0-S)] .LE. R8
C           .LT. SUM FROM {S=0 TO S0} [G4(S)*G4(I0-S)]
C
C    LET R4 = R8 - SUM FROM {S=0 TO S0-1} [G4(S)*G4(I0-S)]
C    THEN    0 .LE. R4 .LT. G4(S0)*G4(I0-S0)
C
C    R4 IS THE OFFSET INTO THE SUBSET S0.
C
C   NOW FIND THE INTEGER S0
C
      R4 = R8
        DO 110 S=1,I0+1
          R4 = R4 - G4(S-1)*G4(I0 - (S-1))
          IF(R4.LT.0) THEN
            S0=S-1
            R4 = R4 + G4(S-1)*G4(I0-(S-1))
            GO TO 120
          ENDIF
  110   CONTINUE
  120 CONTINUE
C.....
C   DECOMPOSE R4 INTO R41 AND R42 WITH 0 .LE. R41 .LE. G4(S0)-1
C     AND  0 .LE. R42 .LE. G4(I0-S0)-1 SO THAT
C     R4 = R42*G4(S0) + R41
C
C    R41 IS THE OFFSET INTO THE SET OF 4-TUPLES {m1,m2,m3,m4}
C         OF WEIGHT S0
C    R42 IS THE OFFSET INTO THE SET OF 4-TUPLES {m5,m6,m7,m8}
C         OF WEIGHT I0-S0
C
      R42 = R4/G4(S0)
      R41 = R4 - R42*G4(S0)
C
C   FIND {m1,m2,m3,m4} FROM R41 AND S0
      CALL ENCOD4(M,G2,S0,R41,M1,M2,M3,M4)
C
C   FIND {m5,m6,m7,m8} FROM R42 AND I0-S0
      CALL ENCOD4(M,G2,I0-S0,R42,M5,M6,M7,M8)
C
C       WRITE(*,*) ' '
C       WRITE(*,*) ' '
C       WRITE(*,*) 'ENCODING RESULTS FOR X = ',X
C       WRITE(*,130) I0,Z8(I0),R8,R4,S0
```



```
C130    FORMAT(' I0 = ',I5,'  Z8(I0) = ',I8,'  R8 = ',I7,'  R4 = ',I5,
C     +    '  S0 = ',I5)
C       WRITE(*,131) R41,R42
C131    FORMAT(' R41 = ',I4,'  R42 = ',I4)
C       WRITE(*,132) M1,M2,M3,M4,M5,M6,M7,M8
C132    FORMAT(' {M1,M2,M3,M4,M5,M6,M7,M8} = {',8I4,' }')
C
C  COMPUTE RING HISTOGRAM
C
        NCOUNT(M1,1) = NCOUNT(M1,1)+1
        NCOUNT(M2,2) = NCOUNT(M2,2)+1
        NCOUNT(M3,3) = NCOUNT(M3,3)+1
        NCOUNT(M4,4) = NCOUNT(M4,4)+1
        NCOUNT(M5,5) = NCOUNT(M5,5)+1
        NCOUNT(M6,6) = NCOUNT(M6,6)+1
        NCOUNT(M7,7) = NCOUNT(M7,7)+1
        NCOUNT(M8,8) = NCOUNT(M8,8)+1
 9999    CONTINUE
      WRITE(*,*) ' '
      WRITE(*,*) '        SHELL HISTOGRAM'
        WRITE(*,*) ' '
        DO 9997 I=0,M-1
        WRITE(*,9998) I,(NCOUNT(I,J),J=1,8)
 9997   WRITE(1,9998) I,(NCOUNT(I,J),J=1,8)
 9998   FORMAT(' ',I3,8I8)
C*****************************************************************
C
C  DECODE {M1,M2,M3,M4,M5,M6,M7,M8} INTO X
C.....
C   FIRST USE {M1,M2,M3,M4}
C
C    FIND T0 AND R2I1
      T0 = M1+M2
      IF(T0.LE.M-1) THEN
        R2I1 = M1
      ELSE
        R2I1 = M-1-M2
      ENDIF
C    FIND S0-T0 = S0MT0 AND R2I2
      S0MT0 = M3+M4
      IF(S0MT0.LE.M-1) THEN
        R2I2 = M3
      ELSE
        R2I2 = M-1-M4
      ENDIF
C    FIND S0 = M1+M2+M3+M4
      S0 = T0 + S0MT0
C    FIND R2I
      R2I = R2I2*G2(T0) + R2I1
C
C    FIND R41 = R2I + SUM FROM {T=0 TO T0} [G2(T)*G2(S0-T)
      R41 = R2I
      DO 140 T=0,T0-1
  140   R41 = R41 + G2(T)*G2(S0-T)
C.....
C   THEN USE {M5,M6,M7,M8}
C
```

```
C    FIND T0 AND R2I1
     T0 = M5+M6
     IF(T0.LE.M-1) THEN
       R2I1 = M5
     ELSE
       R2I1 = M-1-M6
     ENDIF
C    FIND (I0-S0)-T0 = I0S0T0 AND R2I2
     I0S0T0 = M7+M8
     IF(I0S0T0.LE.M-1) THEN
       R2I2 = M7
     ELSE
       R2I2 = M-1-M8
     ENDIF
C    FIND I0-S0 = M5+M6+M7+M8
     I0MS0 = T0 + I0S0T0
C    FIND R2I
     R2I = R2I2*G2(T0) + R2I1
C
C    FIND R42 = R2I + SUM FROM {T=0 TO T0} [G2(T)*G2(I0-S0-T)
     R42 = R2I
     DO 150 T=0,T0-1
150   R42 = R42 + G2(T)*G2(I0MS0-T)
C.....

C
     R4 = R42*G4(S0) + R41
C    FIND I0 = S0 + (I0-S0)
     I0 = S0 + I0MS0
C
C    FIND R8 = R4 + SUM FROM {S=0 TO S0-1} [G4(S)*G4(I0-S)]
C
     R8 = R4
     DO 160 S=0,S0-1
160   R8 = R8 + G4(S)*G4(I0-S)
C
     XOUT = R8 + Z8(I0)
C      WRITE(*,*) ' '
C      WRITE(*,*)'DECODER OUTPUT'
C      WRITE(*,*) 'XOUT =',XOUT
     END


     SUBROUTINE ENCOD4(M,G2,S,R4I,M1,M2,M3,M4)
C
C    THIS SUBROUTINE COMPUTES THE 4-TUPLE {M1,M2,M3,M4} WITH
C    WEIGHT S AND RESIDUE R4I.
C
     INTEGER*4 G2(0:200),S,R4I,T,T0,R2I,R2I1,R2I2
C
C    THE NUMBER OF 4-TUPLES WITH WEIGHT S IS
C      G4(S) = SUM FROM {T=0 TO S} [G2(T)*G2(S-T)]
C
C    PARTITION THE 4-TUPLES INTO S+1 SUBSETS CONSISTING OF THE
C    CONCATENATION OF 2-TUPLES {M1,M2}{M3,M4} WITH WEIGHT{M1,M2} = T
C    AND WEIGHT{M3,M4} = S-T  FOR T=0,1,...,S.  THE SUBSET
C    CORRESPONDING TO RESIDUE R4I IS T0, WHICH IS THE LARGEST
```



```
C    INTEGER SUCH THAT
C      SUM FROM {T=0 TO T0-1} [G2(T)*G2(S-T)] .LE. R4I
C           .LT. SUM FROM {T=0 TO T0} [G2(T)*G2(S-T)]
C
C    LET R2I = R4I - SUM FROM {T=0 TO T0-1} [G2(T)*G2(S-T)]
C    THEN   0 .LE. R2I .LE. G2(T0)*G2(S-T0) - 1
C
C    NOW FIND THE INTEGER T0
C
     R2I= R4I
      DO 10 T=1,S+1
       R2I= R2I- G2(T-1)*G2(S-(T-1))
       IF(R2I.LT.0) THEN
         T0=T-1
         R2I= R2I+ G2(T-1)*G2(S-(T-1))
         GO TO 20
       ENDIF
 10    CONTINUE
20    CONTINUE
C.....
C  DECOMPOSE R2I INTO R2I1 AND R2I2 WITH 0 .LE. R2I1 .LE. G2(T0)-1
C  AND  0 .LE. R2I2 .LE. G2(S-T0)-1 SO THAT
C  R2I = R2I2*G2(T0) + R2I1
C
C  R2I1 IS THE OFFSET INTO THE SET OF 2-TUPLES {m1,m2} OF WEIGHT T0
C  R2I2 IS THE OFFSET INTO THE SET OF 2-TUPLES {m3,m4} OF WEIGHT S-T0
C
     R2I2 = R2I/G2(T0)
     R2I1 = R2I - R2I2*G2(T0)
C.....
C
C  DETERMINE {M1,M2,M3,M4} ACCORDING TO THE RULES ON THE BOTTOM
C  OF PAGE 9 OF THE CCITT MOTOROLA REPORT "SIGNAL MAPPING AND
C  SHAPING FOR V.FAST"
C
     IF(T0.LE.M-1) THEN
       M1 = R2I1
       M2 = T0 - R2I1
     ELSE
       M1 = T0 - (M-1) + R2I1
       M2 = M-1 - R2I1
     ENDIF
C
     IF(S-T0.LE.M-1) THEN
       M3 = R2I2
       M4 = S - T0 - R2I2
     ELSE
       M3 = S - T0 - (M-1) + R2I2
       M4 = M-1 - R2I2
     ENDIF
     RETURN
     END
```

Chapter 9

THE FOUR DIMENSIONAL CONSTELLATION USED BY ITU-T V.34 MODEMS

Modems conforming to the ITU-T Recommendation V.34 [34] have the option of using a 16, 32, or 64-state trellis code. Each code uses a signal constellation selected from a four-dimensional (4D) lattice. This chapter presents the mathematical structure of this 4D constellation and describes how input bits are mapped to constellation points. The constellation is formed by partitioning the 2D constellation into 8 subsets and then combining pairs of 2D subsets into 16 4D subsets. The resulting 4D constellation is invariant to 90° rotations. The lattice partition chains for the 2D and 4D constellations are presented.

9.1 The 2D Constellation and its Partitioning

9.1.1 Generating the 2D Constellation by 90 Degree Rotations of $4\mathbf{Z}^2 + (1, 1)$

The 2D constellations for different V.34 options consist of subsets of the translated 2D integer lattice $2\mathbf{Z}^2 + (1, 1)$. The subset with the largest required size is called the *superconstellation* and contains 1664 points. One quarter of the superconstellation is shown in Figure 9.1. It consists of 416 points from the translated sublattice $4\mathbf{Z}^2 + (1, 1)$. The points are labeled from 0 to 415 so that the magnitude of a point is a nondecreasing function of the label. If two or more points have the same magnitude, the point with the greatest vertical component has the lowest label. All 1664 points of the superconstellation can be generated by 90° clockwise rotations of this quarter constellation. This property helps in making the trellis codes 90° rotationally invariant.

It will now be shown that 90° rotations of the quarter constellation generate all the constellation points. Let the 90° clockwise rotation of a 2D point (x, y) be designated by $\Re(x, y)$. Then

$$\Re(x, y) = (y, -x) \tag{9.1}$$

	-43	-39	-35	-31	-27	-23	-19	-15	-11	-7	-3	1	5	9	13	17	21	25	29	33	37	41	45
45										408	396	394	400	414									
41							398	375	349	339	329	326	335	347	359	386							
37					412	371	340	314	290	279	269	265	273	281	302	322	353	390					
33				401	357	318	282	257	236	224	216	212	218	228	247	270	298	337	378				
29			406	350	306	266	234	206	185	173	164	162	170	181	197	220	253	288	327	379			
25			360	310	263	226	193	165	146	133	123	121	125	137	154	179	207	242	289	338	391		
21		384	324	277	229	189	156	131	110	96	87	83	92	100	117	140	172	208	254	299	354		
17		355	294	243	201	160	126	98	79	64	58	54	62	71	90	112	141	180	221	271	323	387	
13	392	330	274	222	177	135	102	77	55	41	35	31	37	48	65	91	118	155	198	248	303	361	
9	380	316	255	203	158	119	84	60	39	24	17	15	20	30	49	72	101	138	182	230	283	348	415
5	367	304	244	194	148	108	75	50	28	13	6	4	8	21	38	63	93	127	171	219	275	336	402
1	362	296	238	186	142	103	69	43	22	9	1	0	5	16	32	56	85	122	163	213	267	328	395
-3	365	300	240	190	144	106	73	45	25	11	3	2	7	18	36	59	88	124	166	217	272	331	397
-7	372	307	251	199	152	113	80	52	33	19	12	10	14	26	42	66	97	134	174	225	280	341	409
-11	388	320	261	210	167	128	94	67	47	34	27	23	29	40	57	81	111	147	187	237	291	351	
-15	410	343	284	232	183	149	115	89	68	53	46	44	51	61	78	99	132	168	209	258	315	376	
-19		369	311	259	214	175	139	116	95	82	74	70	76	86	104	129	157	195	235	285	342	399	
-23		403	345	292	249	205	176	150	130	114	107	105	109	120	136	161	191	227	268	319	373		
-27			382	332	287	250	215	184	169	153	145	143	151	159	178	202	231	264	308	358	413		
-31				377	333	293	260	233	211	200	192	188	196	204	223	245	278	312	352	404			
-35					383	346	313	286	262	252	241	239	246	256	276	295	325	363	407				
-39						405	370	344	321	309	301	297	305	317	334	356	385						
-43								411	389	374	366	364	368	381	393								

Figure 9.1. One Quarter of Points in the Superconstellation

Let $\Lambda_0 = 4\mathbf{Z}^2 + (1,1)$. Consider the point $(4n_1+1, 4n_2+1)$ in Λ_0 where n_1 and n_2 are integers. Its 90° clockwise rotation is

$$\Re(4n_1+1, 4n_2+1) = (4n_2, -4n_1) + (1,-1) \in 4\mathbf{Z}^2 + (1,-1) \quad (9.2)$$

but

$$4\mathbf{Z}^2 + (1,-1) = 4\mathbf{Z}^2 + (1,-1) + (0,4) = 4\mathbf{Z}^2 + (1,1) + (0,2) \quad (9.3)$$

Thus the rotated subset is

$$\Lambda_1 = \Re\Lambda_0 = \Lambda_0 + (0,2) \quad (9.4)$$

The rotation of the sum of two vectors is the sum of the individually rotated vectors so the 90° clockwise rotation of Λ_1 or 180° clockwise rotation of Λ_0 is

$$\Lambda_2 = \Re\Lambda_1 = \Re\Lambda_0 + \Re(0,2) = [\Lambda_0 + (0,2)] + (2,0) = \Lambda_0 + (2,2) \quad (9.5)$$

The 90° clockwise rotation of Λ_2 is

$$\Lambda_3 = \Re\Lambda_2 = \Re\Lambda_0 + \Re(2,2) = [\Lambda_0 + (0,2)] + (2,-2) = \Lambda_0 + (2,0) \quad (9.6)$$

The original subset Λ_0 and its three rotations correspond to the four cosets of $4\mathbf{Z}^2$ in the partition chain $2\mathbf{Z}^2/4\mathbf{Z}^2$ and are all distinct.

Let the *parity* of a V.34 2D constellation point be defined to be *even* when the sum of its components is divisible by 4, and to be *odd* when the sum is divisible by 2 but not by 4. This nomenclature is motivated by the fact that when the constellation is scaled down by a factor of 2, the sum of components of an even parity point is an even integer and the sum of components of an odd parity point is an odd integer. A 90° rotation changes the parity from even to odd and *vice versa*. Consider the point $(x,y) = (2n_1+1, 2n_2+1)$ where n_1 and n_2 are integers. The sum of its components is $p_0 = x+y = 2(n_1+n_2+1)$. The parity is odd if and only if n_1 and n_2 are both even or both odd integers. Otherwise the parity is even. The sum of the components of the rotated point $\Re(x,y) = (y,-x)$ is $p_1 = 2(n_2-n_1)$ and the parity is even if and only if n_1 and n_2 are both even or both odd and otherwise the parity is odd. Thus the parity of the rotated point is the opposite of the original parity. This parity change property of 90° rotations will be used in partitioning the 4D constellation which consists of the concatenation of a pair of 2D constellations.

9.1.2 Partitioning the 2D Constellation into 8 Subsets

The 4D codes for V.34 require the 2D constellation to be partitioned into 4 or 8 subsets. The partitioning is based on the lattice partition chain

$$2\mathbf{Z}^2/2R\mathbf{Z}^2/4\mathbf{Z}^2/4R\mathbf{Z}^2 \quad (9.7)$$

Each subpartition in the chain has order 2 so the order of $2\mathbf{Z}^2/4R\mathbf{Z}^2$ is 8. The constellation points are formed by translating these lattice points by $(1,1)$. The eight subsets can be represented by the binary labels $(J_2J_1J_0)$ as shown in Table 9.1. In some CCITT (now ITU-T) working papers, they were also represented by the letters a, b, c, d, A, B, C, and D. A lattice formula for the subset corresponding to a binary label is

$$\Lambda(J_2J_1J_0) = 4R\mathbf{Z}^2 + [J_2 \oplus (J_0 \cdot \bar{J}_1)](4,0) + J_1(2,2) + J_0(0,2) + (1,1) \quad (9.8)$$

These representations are shown in Table 9.1. Since (4,4) and (8,0) are points in $4R\mathbf{Z}^2$, they can be added to any of the representations to give an equivalent representation. A small portion of $2\mathbf{Z}^2 + (1,1)$ with the subset labels is shown in Figure 9.2.

Table 9.1. Labels for the 8 Subsets of the 2D Constellation

Binary Label $J_2J_1J_0$	*Letter Label*	*Lattice Representation*
000	a	$4R\mathbf{Z}^2 + (0,0) + (1,1)$
001	b	$4R\mathbf{Z}^2 + (4,2) + (1,1)$
010	c	$4R\mathbf{Z}^2 + (2,2) + (1,1)$
011	d	$4R\mathbf{Z}^2 + (2,4) + (1,1)$
100	A	$4R\mathbf{Z}^2 + (4,0) + (1,1)$
101	B	$4R\mathbf{Z}^2 + (0,2) + (1,1)$
110	C	$4R\mathbf{Z}^2 + (6,2) + (1,1)$
111	D	$4R\mathbf{Z}^2 + (6,4) + (1,1)$

The complete partition tree with the translation vector $(1,1)$ omitted is illustrated in Figure 9.3. Notice that all subsets represented by lower case letters have $J_2 = 0$ and points represented by upper case letters have $J_2 = 1$. Each lower case subset is changed to the corresponding upper case subset by changing J_2 from 0 to 1. Also, each upper case subset is the corresponding lower case subset translated by $(4,0)$.

The union of the "a" and "A" points is

$$\mathcal{A} = \{\mathrm{a}\} \cup \{\mathrm{A}\} = 4\mathbf{Z}^2 + (1,1) \quad (9.9)$$

The 2D quarter superconstellation is a set of 416 points taken from $\mathcal{A}$. Also,

$$\mathcal{B} = \{\mathrm{b}\} \cup \{\mathrm{B}\} = 4\mathbf{Z}^2 + (1,1) + (0,2) \quad (9.10)$$

$$\mathcal{C} = \{\mathrm{c}\} \cup \{\mathrm{C}\} = 4\mathbf{Z}^2 + (1,1) + (2,2) \quad (9.11)$$

$$\mathcal{D} = \{\mathrm{d}\} \cup \{\mathrm{D}\} = 4\mathbf{Z}^2 + (1,1) + (2,0) \quad (9.12)$$

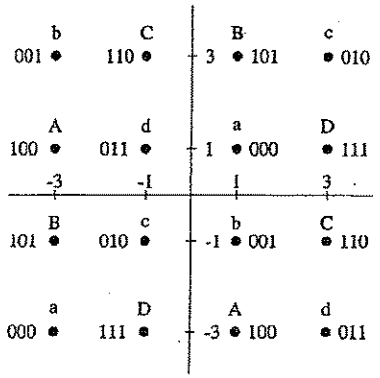


Figure 9.2. Labeling of the 2D Constellation Points

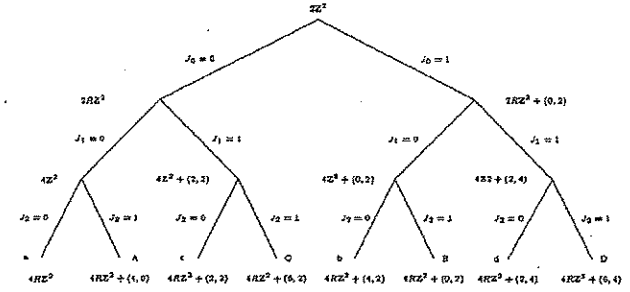


Figure 9.3. The 2D Partition Tree

The subsets have the following properties under 90° clockwise rotations:

$$\underset{000}{\mathrm{a}} \rightarrow \underset{001}{\mathrm{b}} \rightarrow \underset{010}{\mathrm{c}} \rightarrow \underset{011}{\mathrm{d}} \rightarrow \underset{000}{\mathrm{a}} \quad (9.13)$$

$$\underset{100}{\mathrm{A}} \rightarrow \underset{101}{\mathrm{B}} \rightarrow \underset{110}{\mathrm{C}} \rightarrow \underset{111}{\mathrm{D}} \rightarrow \underset{100}{\mathrm{A}} \quad (9.14)$$

$$\mathcal{A} \rightarrow \mathcal{B} \rightarrow \mathcal{C} \rightarrow \mathcal{D} \rightarrow \mathcal{A} \quad (9.15)$$

EXHIBIT E
Part 4

9.1.3 A Method for Determining the Binary Subset Label from the Coordinates of a 2D Point

The combined precoding and trellis coding scheme for V.34 will be presented in Chapter 10. Finding the input to the trellis encoder involves finding the subset binary labels for 2D constellation points. A simple method for finding these labels will now be presented.

Let (x,y) be a 2D constellation point. First translate this point to a lattice point by forming

$$(x_0, y_0) = [(x, y) - (1, 1)]/2 \tag{9.16}$$

According to (9.8)

$$(x_0, y_0) \in 2RZ^2 + [J_2 \oplus (J_0 \cdot \bar{J}_1)](2, 0) + J_1(1, 1) + J_0(0, 1) \tag{9.17}$$

The sum of the components of any point in $2RZ^2$ is some even number $2n$. Thus,

$$x_0 + y_0 = 2n + 2[J_2 \oplus (J_0 \cdot \bar{J}_1)] + 2J_1 + J_0 \tag{9.18}$$

This sum is even if $J_0 = 0$ and is odd if $J_0 = 1$. Let the function $\mathrm{mod}(a, b)$ be defined to be

$$\mathrm{mod}(a, b) = \text{remainder when } a \text{ is divided by } b \tag{9.19}$$

Then the rule for finding J_0 is

$$J_0 = \mathrm{mod}(x_0 + y_0, 2) \tag{9.20}$$

The x component of any point in $2RZ^2$ must be some even integer $2n_1$. Therefore, it follows from (9.17) that

$$x_0 = 2n_1 + 2[J_2 \oplus (J_0 \cdot \bar{J}_1)] + J_1 \tag{9.21}$$

which is even or odd depending on whether J_1 is even or odd. Therefore,

$$J_1 = \mathrm{mod}(x_0, 2) = \text{lsb of } x_0 \tag{9.22}$$

Once J_0 and J_1 have been determined, their effects can be subtracted out to form

$$(x_1, y_1) = \frac{(x_0, y_0) - J_1(1, 1) - J_0(0, 1)}{2}$$

$$\in \frac{2RZ^2 + [J_2 \oplus (J_0 \cdot \bar{J}_1)](2, 0)}{2}$$
$$= RZ^2 + [J_2 \oplus (J_0 \cdot \bar{J}_1)](1, 0) \tag{9.23}$$

The sum of the coordinates of any point in RZ^2 must be some even number $2n_3$. Therefore,

$$x_1 + y_1 = 2n_3 + [J_2 \oplus (J_0 \cdot \bar{J}_1)] \tag{9.24}$$

so

$$[J_2 \oplus (J_0 \cdot \bar{J}_1)] = \mathrm{mod}(x_1 + y_1, 2) \tag{9.25}$$

and

$$J_2 = \mathrm{mod}(x_1 + y_1, 2) \oplus (J_0 \cdot \bar{J}_1) \tag{9.26}$$

9.2 Framing

V.34 modems can operate at data rates ranging from 2400 to 33600 bits per second in increments of 2400 bps and can use six different symbol rates ranging from 2400 to 3429 symbols/sec. In addition, there is a choice of two different carrier frequencies for each symbol rate except just one for 3429. This wide range of bit and symbol rates requires a sophisticated method for mapping data bits to constellation points and in many instances a fractional number of bits per symbol is required. A brief overview of the V.34 data framing method is presented in this section to give a feeling for the complexity involved. See Recommendation V.34 [34] for the complete details.

The frame structure is illustrated in Figure 9.4. The most basic element in the frame structure is the 2D QAM symbol. Pairs of 2D symbols are concatenated to form 4D symbols. In the V.34 recommendation, the pair of 2D symbols making up a 4D symbol are indexed by $k = 0$ or 1. The 2D symbol for $k = 0$ occurs first in time. Then four 4D symbols are concatenated to form a *mapping frame*. The 4D symbols in a mapping frame are indexed by $j = 0$ to 3. The mapping frame consists of eight 2D symbols or 16 dimensions and shell mapping is performed on these frames. Next $P = 12, 14, 15$, or 16 mapping frames are concatenated to form a *data frame* which can be 35 or 40 ms depending on the symbol rate. Finally, $J = 7$ or 8 data frames are concatenated to form a *superframe* which always has a duration of 280 ms.

An integer number of bits are transmitted in each data frame. A V.34 modem has a primary data channel for normal high speed data and a low speed auxiliary channel that is usually used for control and network management functions. Let the sum of the primary and auxiliary data rates be R. Then the number of bits in a data frame is

$$N = R \times 280 \times 10^{-3}/J \tag{9.27}$$

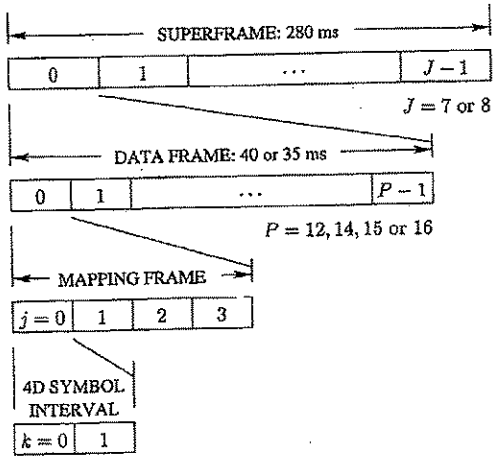


Figure 9.4. V.34 Symbol Framing Scheme

The parameters R and J are chosen to make N an integer.

Since there are P mapping frames in a data frame, the number bits in a mapping frame must be N/P. However, this is not necessarily an integer. Therefore, the number of bits in a mapping frame is switched between b (high frame) and $b-1$ (low frame) so that the total number of bits in a data frame is N. The value of b is the smallest integer not less than N/P, that is,

$$b = \lceil N/P \rceil \tag{9.28}$$

Let r be the number of high mapping frames in a data frame. Then the number of bits in a data frame is

$$N = rb + (P - r)(b - 1) \tag{9.29}$$

Solving for r, we find that the number of high frames must be

$$r = N - (b - 1)P \quad \text{with } 1 \le r \le P \tag{9.30}$$

The low and high frames are selected by switching pattern (SWP) with period P. See the V.34 recommendation [34] for the complete details.

The low mapping frames are formed by replacing one of the data bits entering the 4D mapper by a 0 as shown in Figure 9.5. For a very few low data rate cases, the number of shell mapping bits is $K = 0$ and the 0 is multiplexed into the bit $I_{i,j,3}$. For all other cases ($K > 0$), the 0 is multiplexed into the $S_{i,K}$ bit. The multiplexing is controlled by the SWP switching pattern.

The auxiliary channel bits are multiplexed with the primary channel bits with either 7 or 8 auxiliary channel bits transmitted per data frame depending on the symbol rate. These bits are multiplexed into the $I_{i,j,1}$ bit shown in Figure 9.5 according to a switching pattern called AMP in the V.34 recommendation.

9.3 The 4D Constellation

9.3.1 Mapping Frames and Initial 4D Point Selection

The V.34 mapping from input bits to 4D constellation points involves shell mapping, nonlinear precoding, and a 4D trellis code. Mapping frames are eight 2D channel symbols or, equivalently, four 4D channel symbols in duration. The V.34 recommendation uses the index i to specify the i-th mapping frame in time. The 4D symbols within a mapping frame are indexed by j for $j = 0, 1, 2, 3$. Thus, 4D symbols have the time index

$$m = 4i + j \quad \text{for } j = 0, 1, 2, 3 \tag{9.31}$$

The two 2D symbols within a 4D symbol are indexed by k for $k = 0, 1$, so the total 2D index corresponding to the triple (i, j, k) is

$$n = 2m + k = 8i + 2j + k \quad \text{for } k = 0, 1 \text{ and } j = 0, 1, 2, 3 \tag{9.32}$$

A simplified block diagram of the mapping process from mapping frame bits to 4D constellation points is shown in Figure 9.5. The serial input bits to the modem are first passed thorough a 23-stage, self synchronizing, linear feedback shift register scrambler. Every 8 2D symbols a frame of b input bits is collected from the scrambler, low/high frame multiplexer, and auxiliary channel multiplexer. The frame consists of K bits for the shell mapper, $3 \times 4 = 12$ bits for the rate 3/4 4D trellis encoder, and $8 \times q$ bits where q bits are used to select the point of the quarter constellation in a ring for each of the eight 2D symbols in the mapping frame. Therefore, the total number of bits for the mapping frame is

$$b = K + 12 + 8q \tag{9.33}$$

These bits are arranged as shown in the following array when $K > 0$ which requires that $b > 12$. See [34] for the rare case when $K = 0$. The bit $S_{i,1}$ is the earliest in time and $Q_{i,3,1,q}$ is the latest in time.

$$\begin{array}{llll}
(S_{i,1}, S_{i,2}, \ldots, S_{i,K}) & & & \\
(I_{i,0,1}, I_{i,0,2}, I_{i,0,3}) & (Q_{i,0,0,1}, Q_{i,0,0,2}, \ldots, Q_{i,0,0,q}) & (Q_{i,0,1,1}, Q_{i,0,1,2}, \ldots, Q_{i,0,1,q}) & \\
(I_{i,1,1}, I_{i,1,2}, I_{i,1,3}) & (Q_{i,1,0,1}, Q_{i,1,0,2}, \ldots, Q_{i,1,0,q}) & (Q_{i,1,1,1}, Q_{i,1,1,2}, \ldots, Q_{i,1,1,q}) & \\
(I_{i,2,1}, I_{i,2,2}, I_{i,2,3}) & (Q_{i,2,0,1}, Q_{i,2,0,2}, \ldots, Q_{i,2,0,q}) & (Q_{i,2,1,1}, Q_{i,2,1,2}, \ldots, Q_{i,2,1,q}) & \\
(I_{i,3,1}, I_{i,3,2}, I_{i,3,3}) & (Q_{i,3,0,1}, Q_{i,3,0,2}, \ldots, Q_{i,3,0,q}) & (Q_{i,3,1,1}, Q_{i,3,1,2}, \ldots, Q_{i,3,1,q}) &
\end{array} \tag{9.34}$$

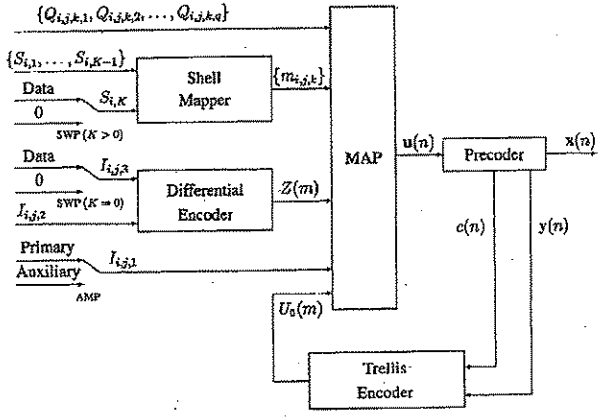


Figure 9.5. Simplified Block Diagram of Mapping Process

The first K bits, $\{S_{i,p}; p = 1, \ldots, K\}$, are used by the Shell Mapper to generate a sequence of 8 2D shell or ring indexes

$$\{m_{i,j,k}\,;\, k = 0, 1 \text{ and } j = 0, 1, 2, 3\} = \{m_{i,0,0}, m_{i,0,1}, m_{i,1,0}, m_{i,1,1}, m_{i,2,0}, m_{i,2,1}, m_{i,3,0}, m_{i,3,1}\} \quad (9.35)$$

Each ring index $m_{i,j,k}$ can have any value in the M element set {0,1,. . .,M-1}. The number of 2D rings, M, depends on the data bit rate, symbol rate, and constellation expansion ratio and is specified by tables in the V.34 recommendation.

Each 2D ring contains 4×2^q points consisting of 2^q points from the quarter superconstellation plus 2^q points from the 90°, 180° and 270° rotations of the quarter superconstellation points. The q bits

$$(Q_{i,j,k,1}, Q_{i,j,k,2}, \ldots, Q_{i,j,k,q})$$

for the 2D symbol at time $n = 8i + 2j + k$ determine the point in ring $m_{i,j,k}$ of the quarter superconstellation. These q bits are considered to be the binary representation of the offset of the point from the first point index in the ring. Thus, the index of the required point in the quarter superconstellation is

$$Q(n) = m_{i,j,k} 2^q + \sum_{\ell=1}^{q} Q_{i,j,k,\ell} 2^{\ell-1} \quad (9.36)$$

Let the pair of 2D points selected from the quarter superconstellation for the 4D symbol at time m be denoted by $\{\mathbf{v}(2m), \mathbf{v}(2m+1)\}$. This will be called the *initial* 4D point. The 2D point $\mathbf{v}(2m)$ will be called the *early symbol* and the point $\mathbf{v}(2m+1)$ the *late symbol*. The three bits $(I_{i,j,1}, I_{i,j,2}, I_{i,j,3})$ along with the bit $U_0(m)$ from the Trellis Encoder are used to convert this pair into the desired final 4D point $\{\mathbf{u}(2m), \mathbf{u}(2m+1)\}$ according to the rules defined in the next subsection. The MAP block in Figure 9.5 represents this transformation. The Precoder and Trellis Encoder blocks are explained in the next chapter.

9.3.2 Mapping the Initial 4D Point Into the Final 4D Point

The method for mapping the initial 4D point $\{\mathbf{v}(2m), \mathbf{v}(2m+1)\}$ into the final 4D point $\{\mathbf{u}(2m), \mathbf{u}(2m+1)\}$ will now be described. The mapping algorithm is show in Figure 9.6 which is a more detailed version of portions of Figure 9.5. The mapping operation converts the initial 4D point into one of 16 translated cosets of $4\mathbf{Z}^4$.

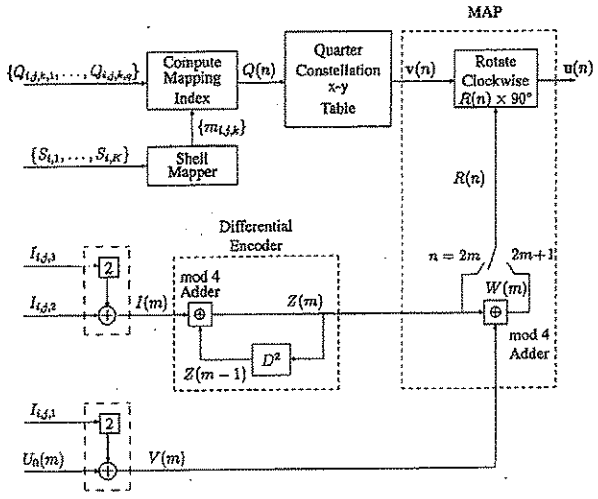


Figure 9.6. Detailed 4D Mapper Block Diagram

The initial 4D point is converted into the final 4D point by rotating the early 2D symbol, $\mathbf{v}(2m)$, clockwise by a multiple of 90° determined by the two input bits $I_{i,j,3}$ and $I_{i,j,2}$, and the late 2D symbol $\mathbf{v}(2m+1)$ by a multiple

determined by all four input bits $I_{i,j,3}$, $I_{i,j,2}$, $I_{i,j,1}$, and $U_0(m)$. The input pair $(I_{i,j,3}, I_{i,j,2})$ is considered to be a 2-bit binary integer with the decimal value $I(m) = 2I_{i,j,3} + I_{i,j,2}$. The value of $I(m)$ specifies the change in rotation of the current early 2D symbol relative to the rotation angle of the early symbol in the previous 4D symbol. Let $(z_1(m), z_2(m))$ be a 2-bit binary number with the decimal value $Z(m) = 2z_1(m) + z_0(m)$. The variable $Z(m-1)$ in the Differential Encoder block is the absolute multiple of 90° that the early symbol of the previous 4D symbol was rotated clockwise. The differential encoder output $Z(m)$ is the absolute multiple of 90° clockwise to rotate the current initial early 2D symbol and can be computed as

$$Z(m) = \text{mod}[Z(m-1) + I(m), 4] \tag{9.37}$$

The input pair $(I_{i,j,1}, U_0(m))$ is considered to be a 2-bit binary number with decimal value

$$V(m) = 2I_{i,j,1} + U_0(m) \tag{9.38}$$

The integer $V(m)$ specifies the multiple of 90° by which the current late 2D symbol $\mathbf{v}(2m+1)$ should be rotated relative to the current early 2D symbol $\mathbf{v}(2m)$. The absolute rotation multiple of 90° for the current late 2D symbol is

$$W(m) = \text{mod}[Z(m) + V(m), 4] \tag{9.39}$$

As shown in Figure 9.6, the initial constellation points are converted into the final points in the MAP block by rotating the early 2D symbols $\mathbf{v}(2m)$ by $Z(m) \times 90°$ clockwise and the late 2D symbols $\mathbf{v}(2m+1)$ by $W(m) \times 90°$ clockwise to give the final 4D point $(\mathbf{u}(2m), \mathbf{u}(2m+1))$.

This mapping procedure partitions $2\mathbf{Z}^4 + (1,1,1,1)$ into 16 cosets of $4\mathbf{Z}^4 + (1,1,1,1)$. The details are displayed in Table 9.2.

9.3.2.1 Rotation Groups

The mapping algorithm can be explained in terms of "rotation groups". The pair of bits, $I_{i,j,1}$ and U_0, partition the 16 4D subsets into four rotation groups as shown in Table 9.3. Each line of the table includes the four 4D subsets that belong to that rotation group and the first entry on the left is the group representative. Notice that if each of the pair of 2D subsets in a 4D subset is rotated 90° clockwise, it becomes the 4D subset circularly to the right within the rotation group. The pair of bits, z_1 and z_0, specifies the 90° clockwise rotation of the group representative.

9.3.3 90° Rotational Invariance of the 4D Constellation

The mapping between input bits and 4D constellation points is invariant to 90° rotations. First, the same shell mapping S bits and uncoded Q bits are assigned to all 90° rotations of 2D points in a mapping frame. Second, the

Table 9.2. Partitioning of $2\mathbf{Z}^4 + (1,1,1,1)$ Induced by the Mapping

z_1	z_0	$I_{i,j,1}$	U_0	4D Subset	Coset of $4\mathbf{Z}^4 + (1,1,1,1)$
0	0	0	0	$\mathcal{A} \times \mathcal{A}$	$4\mathbf{Z}^4 + (1,1,1,1) + (0,0,0,0)$
0	0	0	1	$\mathcal{A} \times \mathcal{B}$	$4\mathbf{Z}^4 + (1,1,1,1) + (0,0,0,2)$
0	0	1	0	$\mathcal{A} \times \mathcal{C}$	$4\mathbf{Z}^4 + (1,1,1,1) + (0,0,2,2)$
0	0	1	1	$\mathcal{A} \times \mathcal{D}$	$4\mathbf{Z}^4 + (1,1,1,1) + (0,0,2,0)$
0	1	0	0	$\mathcal{B} \times \mathcal{B}$	$4\mathbf{Z}^4 + (1,1,1,1) + (0,2,0,2)$
0	1	0	1	$\mathcal{B} \times \mathcal{C}$	$4\mathbf{Z}^4 + (1,1,1,1) + (0,2,2,2)$
0	1	1	0	$\mathcal{B} \times \mathcal{D}$	$4\mathbf{Z}^4 + (1,1,1,1) + (0,2,2,0)$
0	1	1	1	$\mathcal{B} \times \mathcal{A}$	$4\mathbf{Z}^4 + (1,1,1,1) + (0,2,0,0)$
1	0	0	0	$\mathcal{C} \times \mathcal{C}$	$4\mathbf{Z}^4 + (1,1,1,1) + (2,2,2,2)$
1	0	0	1	$\mathcal{C} \times \mathcal{D}$	$4\mathbf{Z}^4 + (1,1,1,1) + (2,2,2,0)$
1	0	1	0	$\mathcal{C} \times \mathcal{A}$	$4\mathbf{Z}^4 + (1,1,1,1) + (2,2,0,0)$
1	0	1	1	$\mathcal{C} \times \mathcal{B}$	$4\mathbf{Z}^4 + (1,1,1,1) + (2,2,0,2)$
1	1	0	0	$\mathcal{D} \times \mathcal{D}$	$4\mathbf{Z}^4 + (1,1,1,1) + (2,0,2,0)$
1	1	0	1	$\mathcal{D} \times \mathcal{A}$	$4\mathbf{Z}^4 + (1,1,1,1) + (2,0,0,0)$
1	1	1	0	$\mathcal{D} \times \mathcal{B}$	$4\mathbf{Z}^4 + (1,1,1,1) + (2,0,0,2)$
1	1	1	1	$\mathcal{D} \times \mathcal{C}$	$4\mathbf{Z}^4 + (1,1,1,1) + (2,0,2,2)$

Table 9.3. Partitioning the 4D Subsets into Rotation Groups

		Rotation within Group $z_1\ z_0$			
		00	01	10	11
Rotation Group $I_{i,j,1}\ U_0$	00	$\mathcal{A} \times \mathcal{A}$	$\mathcal{B} \times \mathcal{B}$	$\mathcal{C} \times \mathcal{C}$	$\mathcal{D} \times \mathcal{D}$
	01	$\mathcal{A} \times \mathcal{B}$	$\mathcal{B} \times \mathcal{C}$	$\mathcal{C} \times \mathcal{D}$	$\mathcal{D} \times \mathcal{A}$
	10	$\mathcal{A} \times \mathcal{C}$	$\mathcal{B} \times \mathcal{D}$	$\mathcal{C} \times \mathcal{A}$	$\mathcal{D} \times \mathcal{B}$
	11	$\mathcal{A} \times \mathcal{D}$	$\mathcal{B} \times \mathcal{A}$	$\mathcal{C} \times \mathcal{B}$	$\mathcal{D} \times \mathcal{C}$

input bit pair $(I_{i,j,3}, I_{i,j,2})$ depends only on the difference in angle between the current early 2D symbol and the early 2D symbol in the previous 4D symbol. The differencing operation cancels any constant rotation angle added to all symbols. Third, the input bit pair $(I_{i,j,1}, U_0(m))$ is determined by the difference in rotation between the current late 2D symbol and current early 2D symbol. Again, a constant rotation is cancelled when the difference is taken. Finally, the trellis codes selected for V.34 all have parity check equations that are transparent to 90° rotations.

The invariance to 90° rotations can also be explained in terms of the rotation groups. A 90° rotation of a 4D subset leaves it in the same rotation group. Therefore, the bits $I_{i,j,1}$ and U_0 are unchanged by this rotation. The bit pair (z_1, z_0) specifies the rotation within a group. This pair of bits is computed by differentially encoding $(I_{i,j,3}, I_{i,j,2})$ modulo 4. Therefore, $(I_{i,j,3}, I_{i,j,2})$ is unchanged by a 90° channel rotation after differential decoding of (z_1, z_0).

9.3.4 Partitioning of the 4D Constellation

The trellis encoders specified for V.34 use the 4D lattice partition chain

$$2\mathbf{Z}^4 \underbrace{/}_{Y_0} 2\mathbf{D}_4 \underbrace{/}_{Y_2} 2R\mathbf{Z}^4 \underbrace{/}_{Y_1} 2R\mathbf{D}_4 \underbrace{/}_{Y_3} 4\mathbf{Z}^4 \underbrace{/}_{Y_4} 4\mathbf{D}_4 \tag{9.40}$$

In the actual constellation, each lattice in this chain is translated by $(1,1,1,1)$. Each subpartition is a two-way partition determined by the corresponding binary variable Y_i so the overall partition $2\mathbf{Z}^4/4\mathbf{D}_4$ is 32-way. The corresponding sequence of minimum squared Euclidean subset distances was shown on page 22 to be

$$d_0^2/d_1^2/d_2^2/d_3^2/d_4^2 = 4/8/8/16/16/32 \tag{9.41}$$

The binary input vectors $(Y_4, Y_3, Y_2, Y_1, Y_0)$ for the trellis encoders are found by slicing the precoder outputs $[y(2m), y(2m+1)]$ shown in Figure 9.5 according to the lattice partition chain. The slicing procedure is described in Subsection 9.3.5. Actually, $Y_0(m)$ is the current parity bit output of the encoder and is computed by the encoder rather than the 4D slicing operation.

When the precoder, which is discussed in Chapter 10, is disabled, the signal $y(n)$ is equal to $u(n)$ and it is reasonable to call the lattice partition chain vector

$$(U_4, U_3, U_2, U_1, U_0) = (Y_4, Y_3, Y_2, Y_1, Y_0) \tag{9.42}$$

Let $Z(m)$ have the binary representation

$$Z(m) = 2z_1(m) + z_0(m) \sim (z_1(m), z_0(m)) \tag{9.43}$$

Remember that V(m) has the representation

$$V(m) = 2I_{i,j,1} + U_0(m) \sim (I_{i,j,1}, U_0(m)) \tag{9.44}$$

Then, the correspondence between the lattice partition variable vector components, except for U_4, and the signals shown entering the MAP box in Figure 9.5 is

$$(U_4, U_3, U_2, U_1, U_0) = (U_4, z_1, z_0, I_{i,j,1}, U_0) \tag{9.45}$$

The bit U_4 (or Y_4 when the precoder is on) does not correspond to a mapper input bit. It is a "phantom" bit generated by the 4D slicer at the trellis encoder input.

The reasons why the input bits generate the desired partition chain will now be presented. First, we will see that U_0 selects the coset in the initial 2-way partition $2\mathbf{Z}^4/2\mathbf{D}_4$. The lattice $\mathbf{D}_4$ is the set of integer 4-tuples whose sum of components is even. Equivalently, it is the set of 4-tuples with an even number of odd components. Therefore, the lattice $2\mathbf{D}_4$ is the set of 4-tuples of even integers that sum to a multiple of 4. The parity of a 4D constellation point is said to be even if the sum of its components is divisible by 4. The 4D parity is odd if the sum is divisible by 2 but not by 4. Equivalently, the 4D parity is even if the sum of components divided by 2 is an even integer and it is odd if the sum of components divided by 2 is an odd integer. We will see that when $U_0 = 0$ the 4D points have even parity and belong to $2\mathbf{D}_4 + (1,1,1,1)$, and when $U_0 = 1$ the points have odd parity and belong to the coset of $2\mathbf{D}_4 + (1,1,1,1)$.

The input variable $V(m)$ determines the rotation of the late 2D symbol in the 4D symbol starting at 4D time index m. When $U_0 = 0$, this rotation is either 0° or 180° depending on the value of $I_{i,j,1}$. A 2D symbol and its 180° rotation have the same parity. Therefore, the late 2D symbol has the same parity as the early 2D symbol when $U_0 = 0$. Suppose the early and late 2D symbols both have even 2D parity. Then the sum of the components for the early symbol is divisible by four and the same for the late symbol. So the sum of all four components is divisible by four. Now suppose the early and late symbols both have odd parity. The sum of components of the early symbol must have the form $4n_1 + 2$ and the sum for the late symbol must have the form $4n_2 + 2$, so the sum of all four components is $4(n_1 + n_2) + 4$ which is divisible by 4. Translating the lattices by $(1,1,1,1)$ adds 4 to the sums and does not change the 4D parity. Therefore, a 4D symbol with $U_0 = 0$ is in $2\mathbf{D}_4 + (1,1,1,1)$ and has even 4D parity. When $U_0 = 1$, the early and late symbols have the opposite parity. Suppose the early symbol has even parity so the sum of its components has the form $4n_1$, and the late symbol has odd parity so the sum of its components has the form $4n_2 + 2$. The sum of all four components is $4(n_1 + n_2) + 2$ which is divisible by 2 but not by four, so the point has odd 4D parity. V.34 4D constellation points must have even or odd 4D parity. Therefore, U_0 selects the coset in the partition of $2\mathbf{Z}^4 + (1,1,1,1)$ into the sublattice $2\mathbf{D}_4 + (1,1,1,1)$ and its coset $2\mathbf{D}_4 + (2,0,0,0) + (1,1,1,1)$.

Notice that any 4-tuple with odd parity could have been used instead of $(2,0,0,0)$ as the coset representative. For example,

$$(0,2,0,0) = (2,0,0,0) + (-2,2,0,0) \tag{9.46}$$

But $(-2,2,0,0)$ has even parity and is an element of $2\mathbf{D}_4$ so

$$\begin{aligned} 2\mathbf{D}_4 + (0,2,0,0) &= 2\mathbf{D}_4 + (-2,2,0,0) + (2,0,0,0) \\ &= 2\mathbf{D}_4 + (2,0,0,0) \end{aligned} \tag{9.47}$$

Now it will be shown that z_0 selects the coset in the second level partition $2\mathbf{D}_4/2R\mathbf{Z}^4$. The 4D lattice $2R\mathbf{Z}^4$ can be defined by the formula

$$2R\mathbf{Z}^4 = (2R\mathbf{Z}^2, 2R\mathbf{Z}^2) \tag{9.48}$$

The lattice $2R\mathbf{Z}^2$ is the set of 2-tuples of even integers whose sum of components is a multiple of 4, that is, have even parity. When the bit $z_0 = 0$, the early 2D symbol selected from the quarter superconstellation is rotated by 0° when $z_1 = 0$ and by 180° when $z_1 = 1$. The quarter 2D superconstellation is a set of points from $\Lambda_0 = 4\mathbf{Z}^2 + (1,1)$ and, according to (9.5), the points in its 180° rotation belong to $\Lambda_2 = 4\mathbf{Z}^2 + (2,2) + (1,1)$. The union of these two sets of points is

$$\{4\mathbf{Z}^2 \cup [4\mathbf{Z}^2 + (2,2)]\} + (1,1) = 2R\mathbf{Z}^2 + (1,1) \tag{9.49}$$

where the result of Example 1.11 scaled up by a factor of two was used. With $U_0 = 0$, the late 2D symbol is rotated 0° if $I_{i,j,1} = 0$ or 180° if $I_{i,j,1} = 1$ with respect to the early symbol. The 180° rotation of $2R\mathbf{Z}^2 + (1,1)$ is also this same lattice. Thus, when $U_0 = 0$ and $z_0 = 0$, the late symbol must also belong to $2R\mathbf{Z}^2 + (1,1)$ and the 4D point belongs to $(2R\mathbf{Z}^2, 2R\mathbf{Z}^2) + (1,1,1,1) = 2R\mathbf{Z}^4 + (1,1,1,1)$. It follows from Example 1.12 that $2R\mathbf{Z}^4$ is a sublattice of $2\mathbf{D}_4$ and $2\mathbf{D}_4/2R\mathbf{Z}^4$ is a two-way partition with

$$2\mathbf{D}_4 = 2R\mathbf{Z}^4 \cup \{2R\mathbf{Z}^4 + (2,0,2,0)\} \tag{9.50}$$

When $z_0 = 1$, similar reasoning shows that the constellation point must belong to the translated lattice

$$2R\mathbf{Z}^4 + (2,0,2,0) + (1,1,1,1) \tag{9.51}$$

which corresponds to the coset of $2R\mathbf{Z}^4$ in $2\mathbf{D}_4$. In other words, z_0 selects one of the two translated lattices on the right hand side of the following equation:

$$\begin{aligned} 2\mathbf{D}_4 + (1,1,1,1) &= \{2R\mathbf{Z}^4 + (1,1,1,1)\} \\ &\quad \cup \{2R\mathbf{Z}^4 + (2,0,2,0) + (1,1,1,1)\} \end{aligned} \tag{9.52}$$

The next level partition $2R\mathbf{Z}^4/2R\mathbf{D}_4$ is controlled by the bit $I_{i,j,1}$. Let $U_0 = z_0 = I_{i,j,1} = 0$. Then $V(m) = 0$ so the late 2D symbol is rotated 0°

relative to the early symbol. When $z_1 = 0$ the early and late symbols are both selected from the unrotated quarter superconstellation and, therefore, the 4D point belongs to the translated 4D lattice

$$\mathbf{A}_0 = (4\mathbf{Z}^2, 4\mathbf{Z}^2) + (1,1,1,1) = 4\mathbf{Z}^4 + (1,1,1,1) \tag{9.53}$$

When $z_1 = 1$, the early and late symbols belong to the 180° rotation of the quarter constellation, and using (9.5) we see the 4D point belongs to

$$\begin{aligned} \mathbf{A}_1 &= (4\mathbf{Z}^2, 4\mathbf{Z}^2) + (2,2,2,2) + (1,1,1,1) \\ &= 4\mathbf{Z}^4 + (2,2,2,2) + (1,1,1,1) \end{aligned} \tag{9.54}$$

Applying the lattice rotation operator R to (9.50) gives

$$\begin{aligned} 2R\mathbf{D}_4 &= 2R\{R\mathbf{Z}^4\} \cup R\{2R\mathbf{Z}^4 + (2,0,2,0)\} \\ &= 4\mathbf{Z}^4 \cup \{4\mathbf{Z}^4 + (2,2,2,2)\} \end{aligned} \tag{9.55}$$

Therefore,

$$\begin{aligned} \mathbf{A}_0 \cup \mathbf{A}_1 &= \{4\mathbf{Z}^4 \cup [4\mathbf{Z}^4 + (2,2,2,2)]\} + (1,1,1,1) \\ &= 2R\mathbf{D}_4 + (1,1,1,1) \end{aligned} \tag{9.56}$$

which is a translated sublattice of $2R\mathbf{Z}^4$. When $I_{i,j,1} = 1$ with $U_0 = z_0 = 0$ similar reasoning shows that points belong to the translated coset $2R\mathbf{D}_4 + (2,2,0,0) + (1,1,1,1)$. Thus, $I_{i,j,1}$ determines the coset in the subpartition $R\mathbf{Z}^4/R\mathbf{D}_4$.

When $z_1 = 0$ the subset $\mathbf{A}_0$ is selected and when $z_1 = 1$ the subset A_1 is selected. In light of (9.56), it is clear that z_1 determines the coset in the subpartition $2R\mathbf{D}_4/4\mathbf{Z}^4$. Finally, the phantom bit U_1 selects the coset in the subpartition $4\mathbf{Z}^4/4\mathbf{D}_4$.

9.3.5 Slicing 4D Points to Partition Chain Binary Variables

To determine the trellis encoder inputs, the precoder output pairs

$$\{y(2m), y(2m+1)\}$$

must be sliced according to the lattice partition chain. This can be done by a table look-up once the subset labels for the early and late 2D symbols are known. An algorithm for determining the 2D subset labels was given in Subsection 9.1.3. The slicing table can be generated by using the mapping algorithm to convert the partition input variables into 4D constellation points and recording the pairs of 2D subset labels. The result is shown in Table 9.4. The following definitions are used in the table.

$$\mathbf{s}(2m) = [J_2(2m), J_1(2m), J_0(2m)] \tag{9.57}$$

and

$$s(2m+1) = [J_2(2m+1), J_1(2m+1), J_0(2m+1)] \tag{9.58}$$

Table 9.4. Table for $[Y_4(m), Y_3(m), Y_2(m), Y_1(m)]$

	$s(2m+1)$							
$s(2m)$	000	001	010	011	100	101	110	111
000	0000	0000	0001	0001	1000	1000	1001	1001
001	0011	0010	0010	0011	1011	1010	1010	1011
010	0101	0101	0100	0100	1101	1101	1100	1100
011	0110	0111	0111	0110	1110	1111	1111	1110
100	1000	1000	1001	1001	0000	0000	0001	0001
101	1011	1010	1010	1011	0011	0010	0010	0011
110	1101	1101	1100	1100	0101	0101	0100	0100
111	1110	1111	1111	1110	0110	0111	0111	0110

Chapter 10

THE COMBINED PRECODING AND TRELLIS CODING SCHEME FOR V.34

This chapter describes the combined nonlinear precoding and trellis coding scheme specified in Recommendation V.34. A block diagram of the system is shown in Figure 10.1. The combined scheme was invented very late in the ITU-T V.34 study committee deliberations. The original idea was presented by R. Laroia [43] during a June 1993 TIA meeting. It was quickly simplified and improved by Bill Betts and R. Laroia of AT&T and, independently, by Paul Cole of General DataComm. The simplified version was documented in a paper by Y. Goldstein of General DataComm, V. Eyuboğlu of Motorola Codex, and S. Olafsson of Rockwell International at a TIA meeting in Newton, Massachusetts in July, 1993 [30]. The precoder is essentially the same as the LTF/Motorola/GDC precoder described in Section 5.2. The new unique idea was to embed the convolutional encoder in a feedback loop with its binary input derived from a local replica of the receiver's prediction error filter output and its output parity check bit fed back to the 4D mapper to select the coset of $2\mathbf{D}_4$ for the input to the precoder. The new precoder's dither signal is confined to a Voronoi region of $2\mathbf{Z}^2$, is independent of the partition tree for the trellis code, and has less power than the dither signal for the LTF precoder in the serial arrangement described in Section 5.2. The precoded sequence applied to the channel is not a trellis sequence. However, the output of the prediction error filter in the receiver is a trellis sequence.

10.1 The Nonlinear Precoder

The purpose of the nonlinear precoder as used in V.34 is to compensate for the noise whitening prediction error filter in the receiver without destroying the constellation shaping resulting from shell mapping. The prediction error filter

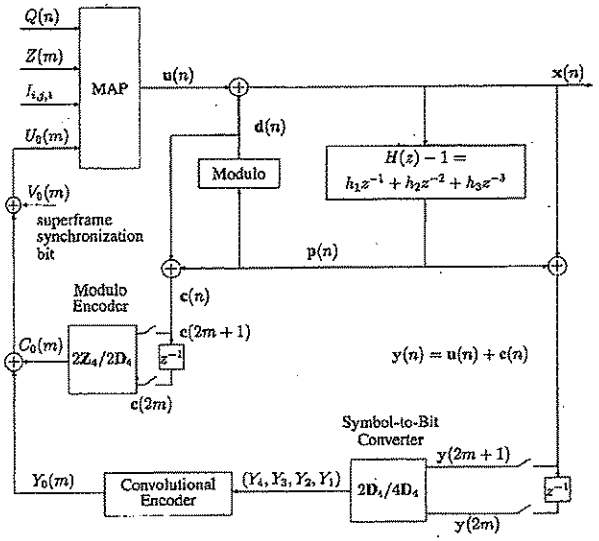


Figure 10.1. Block Diagram of Combined Precoder and Trellis Encoder

specified for V.34 is a 4-tap FIR filter with the transfer function

$$H(z) = 1 + h_1 z^{-1} + h_2 z^{-2} + h_3 z^{-3} \tag{10.1}$$

The filter input is a sequence of 2D symbols which are represented as complex numbers with the real part corresponding to the x value and the imaginary part to the y value. The filter taps are also complex numbers. These taps are computed at the receiver during a segment of the initial training period and sent back to the remote transmitter.

The precoder can be disabled by simply setting the prediction filter coefficients to zero, that is, by letting $h_1 = h_2 = h_3 = 0$. Then $\mathbf{p}(n) = 0$ and is quantized by the modulo box to the point $\mathbf{c}(n) = 0$ in $2\mathbf{Z}^2$ with the quantization error $\mathbf{d}(n) = 0$. Under these conditions, the precoder output is $\mathbf{x}(n) = \mathbf{y}(n) = \mathbf{u}(n)$.

10.1.1 The Precoder Input and Output

The input to the precoder is the sequence of 2D symbols $\mathbf{u}(n)$ generated by the mapping operation described in Chapter 9. These symbols will be considered to be complex numbers. The precoder output $\mathbf{x}(n)$ is formed by adding a dither signal $\mathbf{d}(n)$ to the input, that is

$$\mathbf{x}(n) = \mathbf{u}(n) + \mathbf{d}(n) \tag{10.2}$$

10.1.2 The Prediction Filter Output

The output is passed through the filter $H(z) - 1$ to generate the signal $\mathbf{p}(n)$. The formula for computing $\mathbf{p}(n)$ is

$$\mathbf{p}(n) = h_1\mathbf{x}(n-1) + h_2\mathbf{x}(n-2) + h_3\mathbf{x}(n-3) \tag{10.3}$$

This equation assumes that arithmetic is performed with infinite precision. The V.34 recommendation specifies exactly how the arithmetic should be performed with 16-bit two's complement finite-word-length arithmetic. This is necessary to prevent round-off error accumulation in the receiver. We will see that a crucial fact for implementing the combined precoder and trellis encoder is that $\mathbf{p}(n)$ depends only on the past values of $\mathbf{x}(\cdot)$ and not the present value $\mathbf{x}(n)$. Another important observation is that when the input sequence is rotated by 90°, which corresponds to multiplication by the imaginary constant j, the output is also rotated by 90°.

The output of the prediction error filter $H(z)$ in the receiver can be duplicated in the transmitter by passing $\mathbf{x}(n)$ through $H(z)$. This is easily accomplished by forming the sum

$$\mathbf{y}(n) = \mathbf{x}(n) + \mathbf{p}(n) \tag{10.4}$$

The precoder in conjunction with the Modulo Encoder block operate so that $\mathbf{y}(n)$ is a sequence in the trellis code.

10.1.3 The Modulo Box

The dither signal $\mathbf{d}(n)$ is generated from $\mathbf{p}(n)$ by the Modulo box. Essentially, the Modulo box quantizes $\mathbf{p}(n)$ to the nearest element $\mathbf{c}(n)$ of $2\mathbf{Z}^2$ and forms the dither signal

$$\mathbf{d}(n) = \mathbf{c}(n) - \mathbf{p}(n) \tag{10.5}$$

which is the quantization error. Thus, the dither signal is confined to a Voronoi region of $2\mathbf{Z}^2$.

The Modulo box must also handle the boundary cases where $\mathbf{p}(n)$ is an equal distance from two or four points of $2\mathbf{Z}^2$. This is done in a way that makes the precoder invariant to 90° rotations. Let $p_r(n) = \Re\mathrm{e}\,\mathbf{p}(n)$ and $p_i(n) = \Im m\,\mathbf{p}(n)$

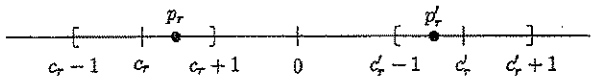


Figure 10.2. Quantization Rules for the Modulo Operator

and similarly with $\mathbf{c}(n)$ and $\mathbf{d}(n)$. Consider the two situations shown in Figure 10.2 for positive and negative $p_r(n)$. When $p_r(n) > 0$, it is quantized to the even integer $c_r(n)$ such that

$$c_r(n) - 1 < p_r(n) \leq c_r(n) + 1 \text{ for } p_r(n) > 0 \tag{10.6}$$

or equivalently

$$-1 \leq d_r(n) < 1 \text{ where } d_r(n) = c_r(n) - p_r(n) \tag{10.7}$$

When $p_r(n) \leq 0$, it is quantized to the even integer $c_r(n)$ such that

$$c_r(n) - 1 \leq p_r(n) < c_r(n) + 1 \text{ for } p_r(n) \leq 0 \tag{10.8}$$

or equivalently

$$-1 < d_r(n) \leq 1 \text{ where } d_r(n) = c_r(n) - p_r(n). \tag{10.9}$$

Notice that a 180° rotation transforms one quantization interval into the other including the open and closed boundaries. The imaginary component is quantized according to the same rules but with the subscript r replaced by i. These rules make the Modulo box transparent to 90° channel rotations.

10.1.4 Why the Precoder is the Inverse of $H(z)$

Based on (10.5) for the dither signal, the precoder can be represented by the simplified block diagram shown in Figure 10.3.

Using the standard rule for a single loop feedback system, it follows that

$$X(z) = [U(z) + C(z)]\frac{1}{H(z)} \tag{10.10}$$

Therefore, the prediction error filter output at the receiver, assuming the adaptive equalizer does a perfect job, is

$$Y(z) = X(z)H(z) = U(z) + C(z) \tag{10.11}$$

or

$$\mathbf{y}(n) = \mathbf{u}(n) + \mathbf{c}(n) \tag{10.12}$$

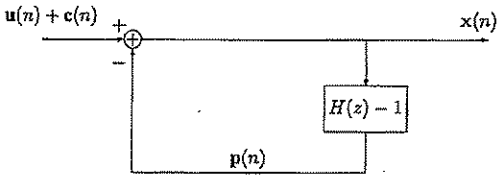


Figure 10.3. Simplified Precoder Block Diagram

The prediction error filter output at the receiver is the precoder input $\mathbf{u}(n)$ modified by the addition of a sequence of symbols $\mathbf{c}(n)$ from $2\mathbf{Z}^2$. In other words, the constellation observed at the output of the prediction error filter in the receiver is expanded beyond the original constellation of the precoder input $\mathbf{u}(n)$ by the addition of points $\mathbf{c}(n)$ from the $2\mathbf{Z}^2$ lattice.

10.2 The Trellis Encoders

Three 4D trellis encoders are included in the V.34 standard: (1) the 16-state, rate 2/3, 4D code invented by L-F Wei at Codex [66], (2) the 32-state, rate 3/4, 4D code invented by R. Williams of British Telecom [6, 7], and (3) the corrected 64-state. rate 4/5, 4D code invented by L-F Wei after he returned to AT&T Bell Labs [3]. Block diagrams of the convolutional encoders are shown in Figures 10.4, 10.5, and 10.6. The 16-state encoder does not use the input bits $Y_3(m)$ and $Y_4(m)$. The 32-state encoder does not use the input bit $Y_3(m)$. The D^2 blocks represent delays of two 2D symbols or one 4D symbol. Each of these encoders is transparent to 90° constellation rotations.

In the V.34 committee deliberations, the 64-state 4D Wei code described in [66] was initially proposed. However, in April 1993 Heegard and Rossin of Cornell and Trott of MIT observed that this code had the flaw of being quasi-catastrophic and had a minimum free squared Euclidean distance of 4, not 5. This flaw was reported to the V.fast study committee by Forney in the May 1993 meeting [55] and Wei presented his new corrected code at this same meeting [3].

The state variables for these encoders are usually taken to be the outputs of the delay elements which are the D^2 blocks. Therefore, $Y_0(m)$ is a state variable for each encoder.

An important property of each of these encoders is that there is a delay of at least one 4D symbol period between the current inputs

$$[Y_4(m), Y_3(m), Y_2(m), Y_1(m)]$$

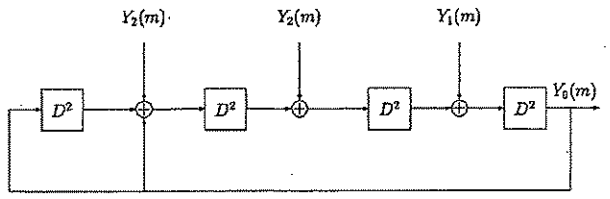


Figure 10.4. The 16-State 4D Wei Convolutional Encoder

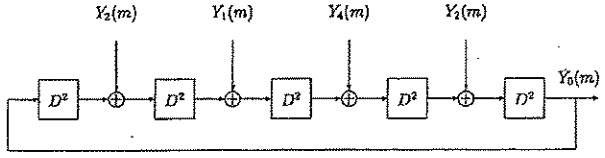


Figure 10.5. The 32-State 4D Williams Convolutional Encoder

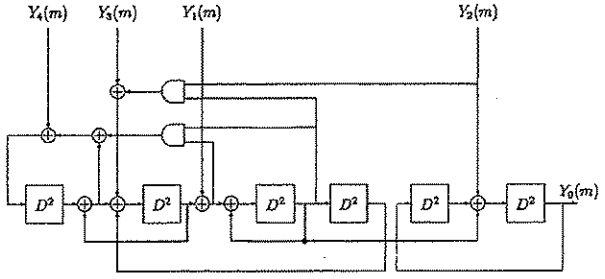


Figure 10.6. The New 64-State 4D Wei Convolutional Encoder

and the current output $Y_0(m)$. This delay is represented by the D^2 block on the right of each encoder. In other words, the current output is not affected by the current inputs, only by the past inputs. The current inputs and encoder state (contents of the delay boxes) are used to compute the next output $Y_0(m+1)$ and the next state. Because of this property, all paths leaving a particular encoder

state in the trellis diagram have the same $Y_0(m)$ bit. Similarly, because $Y_0(m)$ is the input to a D^2 block in each of the encoders, all paths converging on a particular node in the trellis correspond to the same $Y_0(m)$ bit.

Another important property of each of these encoders is that for any given state with $Y_0(m) = 0$, each of the eight 4D subsets with even parity shown in Table 9.2 with $U_0 = 0$ is associated with one of the branches leaving the state. Similarly, for any given state with $Y_0(m) = 1$, each of the eight 4D subsets with odd parity ($U_0 = 1$ in Table 9.2) is associated with one of the branches leaving the state. Also, these same properties hold for branches converging on a state. That is, all the 4D subsets with even parity are associated with branches converging on a state with $Y_0 = 0$, while all the 4D subsets with odd parity are associated with branches converging on a state with $Y_0 = 1$.

Some properties of the V.34 codes as reported in [6] are given in Table 10.1. The row labeled "Minimum Squared Euclidean Distance" is the minimum squared Euclidean distance between trellis paths where d is the minimum distance between points in the 2D constellation. The "Error Coefficient" is the number of paths at the minimum squared Euclidean distance from a given path. The "Effective Coding Gain" is the coding gain based on the increase in distance from uncoded transmission but reduced to take into account the error coefficient.

Table 10.1. Properties of the V.34 Trellis Codes

	Wei 16-State	*Williams 32-State*	Wei 64-State
Minimum Squared Euclidean Distance	$4d^2$	$4d^2$	$5d^2$
Error Coefficient	24	8	144
Effective Coding Gain (dB)	4.2	4.5	4.7
Complexity Relative to V.32 2D Code	1	4	16

An estimate of the computational complexity for a Viterbi decoder for a trellis code is the number of states times the number of branches converging to each state. The rate 2/3, V.32, 8-state 2D trellis code was chosen as a baseline reference. This code has $2^2 = 4$ branches entering each state, so its complexity is $4 \times 8 = 32$. The rate 2/3, 16-state, Wei code also has $2^2 = 4$ branches entering each node, so its 4D complexity is $4 \times 16 = 64$. However, 4D symbols extend over twice the time interval of 2D symbols so the decoder has twice as much time to process each trellis section. Therefore, the 4D complexities are reduced by a factor of two for comparison with 2D codes. Then the 16-state 4D Wei code has the same decoder complexity as the V.32 2D code.

10.3 Viterbi Decoding of 4D Trellis Codes

Viterbi decoding for 4D trellis codes is the same as for 2D codes except that the initial step is to quantize the received pairs of QAM symbols constituting received 4D symbols to ideal points in the 4D subsets. The 4D quantization can be performed by the following two steps:

1 **Quantizing to 2D Subsets**
First quantize the early and late received 2D symbols to each of the eight 2D subsets shown in Table 9.1. For the 16-state code, it is only necessary to quantize to the four subsets $\mathcal{A}, \mathcal{B}, \mathcal{C}$, and $\mathcal{D}$. By "quantizing or slicing to a subset" we mean finding the point in the subset closest to the received point in Euclidean distance. The quantized point along with the squared Euclidean error should be recorded for each subset to be used in following computations.

2 **Quantizing to 4D Subsets**
Next, quantized 2D subsets are combined to form quantized 4D subsets. For each 4D subset compute the squared error by adding the squared errors for the early and late 2D components determined in the first step. The resulting 4D points and their squared errors must be stored for updating the trellis paths. For example, the squared errors for the sixteen 4D subsets listed in Table 9.2 should be computed for the 16-state code.

This process can occasionally lead to 4D points that do not lie in the finite point constellation. This can happen for transmitted points that lie on the boundary of the constellation. One way of handling this is to select a nearby point and let the decoder make an error. A computationally convenient method is to use "infinite grid" slicing where received points are sliced to the infinite lattice without regard to the actual constellation boundaries and live with the occasional errors. It has been found in practice that infinite grid slicing for large constellations causes a negligible decrease in performance.

Next, the cumulative path metrics and survivors can be updated. As an example, consider the 16-state code. Each state has four branches converging on it. The decoder must have knowledge of the four branches converging on each state along with the 4D subsets assigned to each branch. This information can be stored in a table. The decoder must also contain an array containing the cumulative metrics of the survivors to each state. For a given state, the four branch metrics (squared Euclidean errors) for the paths converging on that state should be added to the cumulative path metrics for the states the branches come from. The surviving path to the given state is the one corresponding to the smallest of the four cumulative path metrics to the given state. The surviving cumulative path metric should be stored in the updated path metric table and a pointer to the best previous state should be stored in the trellis record for the given state along with constellation point selected for the branch. The cumulative metric for a previous state is used to update the paths to four next states, so care must be taken not to destroy the previous metrics before they are used. Assuming a software implementation, one solution is to have two cumulative metric arrays. One array contains the metrics for survivors to the previous states and the other is used to store the cumulative metrics for survivors to the current states. The arrays can be swapped at the next decoder iteration with the array for the new metrics in the previous iteration becoming the array for the previous metrics and the new results written into the array for the previous metrics in the previous iteration.

The final step is to trace the estimated trellis path back to the beginning of the trellis storage array to get the delayed estimate of the most likely transmitted constellation point at that time. The state with the smallest cumulative metric in the current cumulative metric array should be used as the starting point for the trace back. The estimated symbol sequence can then be applied to the inverse LTF precoder to determine the sequence at the output of the MAP block shown in Figure 10.1 and the inverse of the MAP performed to estimate the input bit sequence.

10.4 More Details on the Wei 16-State Code

The Wei 16-state 4D trellis code is one of the most widely used trellis codes at the current time. It is one of the three codes included in ITU-T Recommendation V.34 and is used most often by V.34 modems because the decoding computational complexity is much less than for the 32 and 64-state codes. It is included in the ITU-T Recommendation V.90 which uses the V.34 mode in the upstream direction from the client modem to server. It is also included as an option in the new ITU-T Recommendation V.92 in the upstream direction where symbols are one-dimensional PCM levels transmitted at 8000 symbols/sec. In this mode, four consecutive PCM symbols are considered to be a 4D symbol. This code is also specified as an option in asymmetrical digital subscriber line (ADSL) standards using discrete multi-tone modulation (DMT). Because of the importance of this code, additional details are discussed in this section.

10.4.1 Generator and Check Matrices

The 16-state encoder shown in Figure 10.4 has the form of the type 2 direct form realization shown in Figure 3.3. Therefore, the input and output are related by

$$[Y_0(D^2)\ Y_1(D^2)\ Y_2(D^2)] = [Y_1(D^2)\ Y_2(D^2)]\ \mathbf{G}(D^2) \qquad (10.13)$$

where the $\mathbf{G}(D^2)$ is the code generator matrix

$$\mathbf{G}(D^2) = \begin{bmatrix} \dfrac{D^2}{1+(D^2)^3+(D^2)^4} & 1 & 0 \\ \dfrac{(D^2)^2+(D^2)^3}{1+(D^2)^3+(D^2)^4} & 0 & 1 \end{bmatrix} = [\mathbf{P}(D^2)\ \mathbf{I}_{2\times 2}] \qquad (10.14)$$

A check matrix for this code is

$$\begin{aligned} \mathbf{H}(D^2) &= [1\ \mathbf{P}^t(D^2)] \\ &= \begin{bmatrix} 1 & \dfrac{D^2}{1+(D^2)^3+(D^2)^4} & \dfrac{(D^2)^2+(D^2)^3}{1+(D^2)^3+(D^2)^4} \end{bmatrix} \end{aligned} \qquad (10.15)$$

The following more convenient check matrix without feedback can be formed by multiplying each term by the least common multiple of the denominators:

$$\mathbf{H}(D^2) = \begin{bmatrix} 1+(D^2)^3+(D^2)^4 & D^2 & (D^2)^2+(D^2)^3 \end{bmatrix} \qquad (10.16)$$

The parity check equation in the time domain is

$$\begin{aligned} Y_0(m) + Y_0(m-3) + Y_0(m-4) + Y_1(m-1) \\ + Y_2(m-2) + Y_2(m-3) = 0 \end{aligned} \qquad (10.17)$$

where addition is performed modulo 2.

10.4.2 Invariance to 90 Degree Rotations

It was shown in Section 9.3.3 that the input data bits estimated at the receiver remain unchanged when the V.34 4D constellation is rotated by 90° if the bits $I_{i,j,2}$ and $I_{i,j,3}$ are differentially encoded modulo 4. However, this is not enough to make the trellis code invariant to 90° rotations. It also must be shown that the rotated sequence is also a trellis sequence. This will be true if the input bits corresponding to the rotated sequence also satisfy the encoder's parity check equation. As discussed in Section 9.3.3, the bit pair $[Y_1(m), Y_0(m)]$ selects the rotation group and does not change with 90° rotations. The bit pair $[Y_3(m), Y_2(m)]$ selects the rotation group. A 90° clockwise rotation changes this pair, when considered as a two-bit binary number, to

$$[Y_3'(m), Y_2'(m)] = [Y_3(m) + Y_2(m), Y_2(m) + 1] \qquad (10.18)$$

where the additions are performed modulo 2. Substituting these results into the left hand side of the parity check equation (10.17) gives

$$\begin{aligned} Y_0(m) &+ Y_0(m-3) + Y_0(m-4) + Y_1(m-1) + [Y_2(m-2)+1] \\ &\quad + [Y_2(m-3)+1] \\ &= Y_0(m) + Y_0(m-3) + Y_0(m-4) + Y_1(m-1) \\ &\quad + Y_2(m-2) + Y_2(m-3) = 0 \end{aligned} \qquad (10.19)$$

Therefore, rotated sequences are also trellis sequences and the code is invariant to 90° rotations.

10.4.3 The Fundamental Coding Gain

From the encoder block diagram, it is evident that all branches diverging from a particular state all have the same $Y_0(m)$ and use constellation points from the same subset at level 1 in the partition tree. Similarly, all branches converging on a state have the same Y_0 and must use points from the same subset at level 1. The subsets at level 1 are cosets of $2\mathbf{D}_4$. The minimum squared distance in $\mathbf{D}_4$ is 2, so $d^2_{\min} = 8$ for $2\mathbf{D}_4$. Therefore, the squared Euclidean distance between two trellis paths that diverge and remerge more than one stage later must be at least $2 \times 8 = 16$. It can be shown that the minimum squared free Euclidean distance, d^2_f, is greater than 16 because the contribution from branches between the diverging and converging sections is nonzero. Points along the parallel transitions are selected from the coding sublattice $4\mathbf{Z}^2$ which has MSED $d^2_p = 16$ which is less than the MSED for diverging paths. Therefore, the minimum squared Euclidean distance for this code is $d^2_{\min} = 16$.

The redundancy of the coding lattice is $r(2\mathbf{Z}^2) = |\mathbf{Z}^2/2\mathbf{Z}^2| = 4$ and the redundancy of the convolutional encoder is $r(C) = 1$ so the total redundancy of the trellis code is $r(\mathbb{C}) = 5$ and the normalized redundancy is $\rho(\mathbb{C}) = 2.5$. According to (3.98), the fundamental coding gain is

$$\gamma(\mathbb{C}) = 2^{-\rho(\mathbb{C})} d^2_{\min}((\mathbb{C}) = 2^{-2.5} \times 16 = 2\sqrt{2} \quad \text{or} \quad 4.515\ \text{dB} \qquad (10.20)$$

10.4.4 The Original Wei 16-State Convolutional Encoder

Wei used the realization shown in Figure 10.7 for the convolutional encoder of the 16-state 4D code in his paper [66]. This is neither a type 1 nor a type 2 direct form realization. However, the transfer functions from the inputs $Y_1(m)$ and $Y_2(m)$ to the output $Y_0(m)$ are the same as for the V.34 encoder, that is, it has the same generator matrix. The open loop gain for the encoder is $(D^2)^3(1+D^2) = (D^2)^3 + (D^2)^4$. Using the standard single feedback loop transfer function formula and superposition, it follows that

$$\begin{aligned} Y_0(D^2) &= Y_1(D^2)\frac{D^2}{1+(D^2)^3+(D^2)^4} \\ &\quad + Y_2(D^2)\frac{(D^2)^2+(D^2)^3}{1+(D^2)^3+(D^2)^4} \end{aligned} \qquad (10.21)$$

which is exactly the same as for the V.34 16-state encoder.

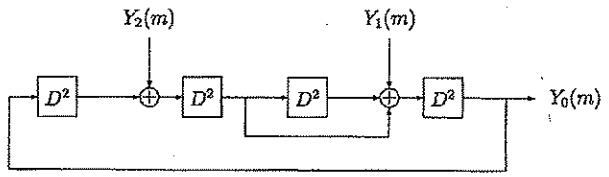


Figure 10.7. The Original Wei 16-State Convolutional Encoder

10.5 Using the Modulo Encoder to Make $\mathbf{y}(n)$ a Trellis Sequence

To understand how the combined precoder and trellis encoder works, we will first consider the case where the precoder is disabled by making $h_1 = h_2 = h_3 = 0$. Then $\mathbf{y}(n) = \mathbf{x}(n) = \mathbf{u}(n)$. Also assume the superframe synchronization bit $V_0(m) = 0$ for all m. The Symbol-to-Bit Converter block converts the 4D symbol $[\mathbf{u}(2m), \mathbf{u}(2m+1)]$ into the lattice partition chain vector

$$[U_4(m), U_3(m), U_2(m), U_1(m)] = [Y_4(m), Y_3(m), Y_2(m), Y_1(m)] \quad (10.22)$$

corresponding to the following partition chain

$$2\mathbf{D}_4 \underbrace{/}_{U_2} 2R\mathbf{Z}^4 \underbrace{/}_{U_1} 2R\mathbf{D}_4 \underbrace{/}_{U_3} 4\mathbf{Z}^4 \underbrace{/}_{U_4} 4\mathbf{D}_4 \quad (10.23)$$

As described in Chapter 9, this can be done by first determining the 2D subset labels $[J_2(2m), J_1(2m), J_0(2m)]$ and $[J_2(2m+1), J_1(2m+1), J_0(2m+1)]$ for $\mathbf{u}(2m)$ and $\mathbf{u}(2m+1)$, respectively, by using the method of Section 9.1.3. These subset labels can then be used in Table 9.4 to look up the 4D partition chain variables. In the usual situation where the trellis encoder is not in a feedback loop, these are the input bits for the encoder at 4D symbol time m. The current output bit $Y_0(m) = U_0(m)$ is not a function of these input bits because of the encoder delay property discussed above. However, they are used, along with the current encoder state, to compute the next encoder output $Y_0(m+1) = U_0(m+1)$. Therefore, the output sequence $\mathbf{u}(n)$ must correspond to a path through the trellis. In other words, the encoder input bits can be determined from the currently transmitted 4D symbol and then these bits can be used as the encoder input to compute the encoder output to use with the next 4D symbol. When the precoder is disabled, the system acts exactly like an open loop system with the MAP box following the convolutional encoder and inputs $I_{i,j,1}$, $Z(m)$, and $Q(n)$.

Now consider the case where the precoder is operating. The situation becomes different because, according to (10.12), $\mathbf{y}(n) = \mathbf{u}(n) + \mathbf{c}(n)$ where $\mathbf{c}(n)$ is an element of $2\mathbf{Z}^2$. Without the correction of the Modulo Encoder block, the sequence $\mathbf{y}(n)$ is not guaranteed to be a trellis sequence because $\mathbf{c}(n)$ is selected by the precoder and not the convolutional encoder. In Chapter 9 the parity of a 4D symbol was defined to be even if the sum of its four components divided by 2 is even, and odd otherwise. The Modulo Encoder computes the parity of the 4D symbol $[\mathbf{c}(2m), \mathbf{c}(2m+1)]$ and sets $C_0(m)$ to 0 for even parity and to 1 for odd parity. That is,

$$C_0(m) = \begin{cases} 0 \text{ if Parity}[\mathbf{c}(2m), \mathbf{c}(2m+1)] \text{ even} \\ 1 \text{ if Parity}[\mathbf{c}(2m), \mathbf{c}(2m+1)] \text{ odd} \end{cases} \quad (10.24)$$

Notice that $[\mathbf{c}(2m), \mathbf{c}(2m+1)]$ alters the parity of $[\mathbf{u}(2m), \mathbf{u}(2m+1)]$ in forming the sum for $\mathbf{y}(n)$ when its parity is odd and leaves it the same when its parity is even.

It does not matter how $[\mathbf{u}(2m), \mathbf{u}(2m+1)]$ is altered by the addition of $[\mathbf{c}(2m), \mathbf{c}(2m+1)]$ to get $[\mathbf{y}(2m), \mathbf{y}(2m+1)]$ as long as this sum has the 4D parity specified by the encoder output bit $Y_0(m)$ because each of the eight 4D subsets with parity corresponding to $Y_0(m)$ is associated with one of the branches leaving that encoder state. Consequently, there is always a trellis path leaving each state with the 4D subset of $[\mathbf{y}(2m), \mathbf{y}(2m+1)]$.

The output bit $Y_0(m)$ was determined by the previous 4D symbol

$$[\mathbf{y}(2m-2), \mathbf{y}(2m-1)]$$

as a function of the Symbol-to-Bit Converter output

$$[Y_4(m-1), Y_3(m-1), Y_2(m-1), Y_1(m-1)]$$

and the state at 4D time $m-1$. The correct parity can be forced by modifying the late 2D symbol $\mathbf{u}(2m+1)$. The following three critical properties of the system make this correction possible:

1 $Y_0(m)$ depends only on the past 4D convolutional encoder inputs.

2 The $U_0(m)$ bit only affects the late symbol $\mathbf{u}(2m+1)$. In addition, the parity of the 4D symbol $[\mathbf{u}(2m), \mathbf{u}(2m+1)]$ is the same as that of $U_0(m)$.

3 The predictor output $\mathbf{p}(n)$ only depends on past $\mathbf{x}$'s.

The sequence of computations required to implement the combined precoder and trellis encoder will now be described in more detail. Once the late symbols $\mathbf{u}(2m-1)$, $\mathbf{x}(2m-1)$, and $\mathbf{y}(2m-1)$ for the previous 4D baud have been computed, it is immediately possible to do the following before it is time to transmit the current early symbol:

1 Slice $[\mathbf{y}(2m-2), \mathbf{y}(2m-1)]$ to a lattice partition chain vector with the Symbol-to-Bit Converter.

2 Compute the new convolutional encoder output $Y_0(m)$.

3 $U_0(m)$ does not affect the early 2D symbol in the current 4D symbol. Therefore, immediately compute the early symbol $\mathbf{u}(2m)$ with the MAP block.

4 Compute $\mathbf{p}(2m)$ from the past $\mathbf{x}$'s using the equation

$$\mathbf{p}(2m) = h_1\mathbf{x}(2m-1) + h_2\mathbf{x}(2m-2) + h_3\mathbf{x}(2m-3) \qquad (10.25)$$

5 Compute $\mathbf{d}(2m)$ with the Modulo box.

6 Compute $\mathbf{c}(2m)$ as

$$\mathbf{c}(2m) = \mathbf{p}(2m) + \mathbf{d}(2m) \qquad (10.26)$$

7 Compute the early channel output symbol $\mathbf{x}(2m)$ as

$$\mathbf{x}(2m) = \mathbf{u}(2m) + \mathbf{d}(2m) \qquad (10.27)$$

8 Compute the encoder input $\mathbf{y}(2m)$ as

$$\mathbf{y}(2m) = \mathbf{x}(2m) + \mathbf{p}(2m) \qquad (10.28)$$

At time $n = 2m$, transmit $\mathbf{x}(2m)$. Then it is possible to determine if the precoder will alter the parity of the current 4D symbol and make the appropriate correction because $\mathbf{c}(2m+1)$ can be predicted. The sequence of steps required before time $2m+1$ when the late symbol must be transmitted are:

1 Predict $\mathbf{p}(2m+1)$ by the equation

$$\mathbf{p}(2m+1) = h_1\mathbf{x}(2m) + h_2\mathbf{x}(2m-1) + h_3\mathbf{x}(2m-2) \qquad (10.29)$$

2 Predict $\mathbf{d}(2m+1)$ by putting $\mathbf{p}(2m+1)$ through the Modulo box.

3 Predict $\mathbf{c}(2m+1)$ as

$$\mathbf{c}(2m+1) = \mathbf{p}(2m+1) + \mathbf{d}(2m+1) \qquad (10.30)$$

4 Compute the parity of $\mathbf{C}_m = [\mathbf{c}(2m), \mathbf{c}(2m+1)]$ by summing the four components and dividing by 2. The parity is even if the result is and even integer and odd if the result is an odd integer. Output $C_0(m) = 0$ for even parity and $C_0(m) = 1$ for odd parity. This function is performed by the Modulo Encoder block.

5 Compute the MAP input $U_0(m)$ for the late symbol as

$$U_0(m) = Y_0(m) \oplus C_0(m) \oplus V_0(m) \qquad (10.31)$$

For the moment, ignore the superframe synchronization bit $V_0(m)$ and assume it is 0. When $\mathbf{C}_m$ has even parity, no compensation is required because adding it to a 4D symbol does not change the symbol's parity. In this case, $U_0(m) = Y_0(m)$. When $\mathbf{C}_m$ has odd parity, $U_0(m)$ is the logical complement of $Y_0(m)$. This complementation reverses the parity change that would have been caused by the precoder in the 4D symbol $[\mathbf{y}(2m), \mathbf{y}(2m+1)]$ and makes its parity the value specified by $Y_0(m)$.

6 Compute the late symbol $\mathbf{u}(2m+1)$ with the MAP block.

7 Compute the late symbol to be transmitted $\mathbf{x}(2m+1)$ as

$$\mathbf{x}(2m+1) = \mathbf{u}(2m+1) + \mathbf{d}(2m+1) \qquad (10.32)$$

8 Compute the late trellis encoder input $\mathbf{y}(2m+1)$ as

$$\mathbf{y}(2m+1) = \mathbf{x}(2m+1) + \mathbf{p}(2m+1) \qquad (10.33)$$

At time $2m+1$, the late symbol must be transmitted. Then, before the start of the next 4D baud, the current 4D symbol $[\mathbf{y}(2m), \mathbf{y}(2m+1)]$ should be sliced to the lattice partition chain vector (Y_4, Y_3, Y_2, Y_1) by the Symbol-to-Bit Converter, and the next encoder output $Y_0(m+1)$ calculated.

10.6 Superframe Synchronization

The V.34 frame structure was discussed in Section 9.2. It is imperative to achieve and maintain superframe synchronization. Each superframe consists of $J = 7$ or 8 data frames; each data frame contains $P = 12$, 14, 15, or 16 mapping frames; and each mapping frame contains four 4D symbols. Therefore, each data frame contains $4P$ 4D symbols. Information for superframe synchronization is introduced into the V.34 signal by setting the superframe synchronization bit $V_0(m)$ to 1 according to a "periodic bit inversion pattern." These 1's are introduced in the 4D symbol intervals at the beginning and middle of each data frame, that is, when the 4D symbol time m is a multiple of $2P$, according to the pattern shown in Table 10.2. In each row of the table, time increases from the left to the right. The left bit in each pair is the value of V_0 in the 4D symbol at the beginning of the data frame and the right bit is the value at the center of the data frame. The period of the frame synchronization pattern is 16 for $J = 8$ and 14 for $J = 7$.

In Section 9.3.2, it was explained that the effect of the $U_0(m)$ bit when it is 1 is to add a 90° clockwise rotation to the late 2D symbol $\mathbf{u}(2m+1)$ relative

Table 10.2. V.34 Framing Bit Pattern

J	Pattern							
8	10	11	01	11	11	11	01	10
7		01	11	01	11	11	11	10

to the early 2D symbol $\mathbf{u}(2m)$. When $V_0(m) = 1$, it logically complements $Y_0(m) \oplus C_0(m)$ and when $V_0(m) = 0$ it causes no change. Therefore, the effect of $V_0(m) = 1$ is to (a) add a 90° clockwise rotation to the late symbol $\mathbf{u}(2m+1)$ when $Y_0(m) \oplus C_0(m) = 0$ or (b) add a 90° counterclockwise rotation to $\mathbf{u}(2m+1)$ when $Y_0(m) \oplus C_0(m) = 1$. In either case, the parity of the 4D symbol transmitted at 4D time m is complemented. These parity changes from the nominal values at the bit inversion times can be detected and used for superframe synchronization at the receiver.

First consider the case when the precoder is disabled so that $\mathbf{y}(n) = \mathbf{u}(n)$. Complementing $U_0(m)$ by setting $V_0(m)$ to 1 rotates $\mathbf{y}(2m+1)$, the late symbol, by 90° clockwise or counterclockwise depending on whether $Y_0(m) = 0$ or 1 and complements the parity of the 4D symbol $[\mathbf{y}(2m), \mathbf{y}(2m+1)]$. However, it is clear from the mapping rules that it does not change the higher order bits specifying the symbol. Therefore, the vector (Y_4, Y_3, Y_2, Y_1) at the input to the convolutional encoder in Figure 10.1 is unchanged by the superframe synchronization bit inversions and the encoder follows the same sequence of states it would have if $V_0(m)$ were always 0. It is a simple matter to account for these rotations of the late 2D symbols in the Viterbi decoder at the receiver. Suppose at a bit inversion time the Viterbi decoder is updating a state for which all the branches entering it have 4D symbols with even parity so that $Y_0 = 0$ for each. However, the encoder has transmitted symbols with $Y_0 = 1$ consisting of the original early symbol and the late symbol rotated 90° clockwise. The decoder can rotate the late symbol 90° counterclockwise and then proceed normally. If a state is being updated for which all the branches entering it have odd 4D parity, the encoder in the transmitter complemented the parity, used $Y_0 = 0$, and rotated the late symbol 90° counterclockwise. In this case, the decoder can rotate the late symbol 90° clockwise and proceed normally.

Now consider the situation when the precoder is operating. The 4D symbols normally transmitted consist of a pair of 2D symbols with the form

$$\mathbf{y}(2m) = \mathbf{u}(2m) + \mathbf{c}(2m) \quad \text{and} \quad \mathbf{y}(2m+1) = \mathbf{u}(2m+1) + \mathbf{c}(2m+1) \tag{10.34}$$

with $\mathbf{u}(\cdot) \in 2\mathbf{Z}^2 + (1,1)$ and $\mathbf{c}(\cdot) \in 2\mathbf{Z}^2$. The transmitted 2D symbols $\mathbf{y}(\cdot)$ belong to one of the subsets $\mathcal{A}, \mathcal{B}, \mathcal{C}$, or $\mathcal{D}$ defined in Section 9.1.2. During a synchronization symbol interval, the early 2D symbol is unchanged and the late

symbol is changed by rotating $\mathbf{u}(2m+1)$ an additional 90° clockwise if $Y_0(m) \oplus C_0(m)$ was 0 before being complemented to 1 or by 90° counterclockwise if $Y_0(m) \oplus C_0(m)$ was 1 before being complemented. This complements the parity of the transmitted 4D symbol. Let the modified $\mathbf{u}(2m+1)$ be denoted by $\mathbf{u}'(2m+1)$. Then the 2D symbols transmitted during a synchronization period are

$$\mathbf{y}(2m) = \mathbf{u}(2m) + \mathbf{c}(2m) \tag{10.35}$$

and

$$\mathbf{y}'(2m+1) = \mathbf{u}'(2m+1) + \mathbf{c}(2m+1) \tag{10.36}$$

Now the effect is not just a simple 90° rotation of the late symbol $\mathbf{y}(2m+1)$. Suppose $\mathbf{y}(2m+1)$ belongs to the 2D subset $\mathcal{X} \in \{\mathcal{A}, \mathcal{B}, \mathcal{C}, \mathcal{D}\}$. It can be shown that the effect of the bit inversion caused by $V_0(m) = 1$ is to transform $\mathbf{y}(2m+1)$ into a point in the subset (a) $\mathcal{X}$ rotated by 90° clockwise in the sequence given by (9.15) if $Y_0(m) = 0$ or (b) $\mathcal{X}$ rotated by 90° counterclockwise if $Y_0(m) = 1$. In addition, the modified 4D symbol has the same value of $[Y_4, Y_3, Y_2, Y_1]$ so the encoder follows the same sequence of states that would have occurred without the bit inversions.

EXAMPLE 10.1 Examples of the Subset Rotation Property

Suppose the received 4D symbol has the form

$$\mathbf{y}(2m)] = \mathbf{u}(2m) + \mathbf{c}(2m) \in \mathcal{A} \tag{10.37}$$

and

$$\mathbf{y}(2m+1) = \mathbf{u}(2m+1) + \mathbf{c}(2m+1) \in \mathcal{A} \tag{10.38}$$

According to Table 9.4, a symbol of this type has the lattice partition bit vector $(Y_4, Y_3, Y_2, Y_1) = (0,0,0,0)$. We know that $\mathbf{c}(2m+1) \in 2\mathbf{Z}^2$ and $\mathcal{A} = 4\mathbf{Z}^2 + (1,1)$. Suppose that $\mathbf{u}(2m)$ and $\mathbf{u}(2m+1)$ both belong to $\mathcal{A}$. In order for $\mathbf{y}(2m)$ and $\mathbf{y}(2m+1)$ both to be in $\mathcal{A}$, it is necessary that $\mathbf{c}(2m)$ and $\mathbf{c}(2m+1)$ be an elements of $4\mathbf{Z}^2$. The pair $[\mathbf{c}(2m), \mathbf{c}(2m+1)]$ has even parity, so $C_0(m) = 0$. Since 4D symbols of the type $\mathcal{A} \times \mathcal{A}$ have even parity, the encoder output bit must be $Y_0(m) = 0$. Complementing this bit rotates $\mathbf{u}(2m+1)$ 90° clockwise to the subset

$$\Re\{\mathcal{A}\} = \mathcal{B} = 4\mathbf{Z}^2 + (1,-1) \tag{10.39}$$

and the modified late 2D symbol $\mathbf{y}'(2m+1) = \mathbf{u}'(2m+1) + \mathbf{c}(2m+1)$ must also belong the $\mathcal{B}$. The resulting 4D symbol at the output of the prediction error filter is of the type $\mathcal{A} \times \mathcal{B}$ which, according to Table 9.4, also has the lattice partition vector $(0,0,0,0)$.

Now suppose $\mathbf{u}(2m+1)$ is in $\mathcal{B}$. For $\mathbf{y}(2m+1)$ to be in $\mathcal{A}$, $\mathbf{c}(2m+1)$ must be in $4\mathbf{Z}^2 + (0,2)$. Then $[\mathbf{c}(2m), \mathbf{c}(2m+1)]$ has odd parity and $C_0(m) = 1$ and

$U_0(m) = Y_0(m) + C_0(m) + V_0(m) = 1$ if $V_0(m) = 0$. At a synchronization time $V_0(m) = 1$ and $U_0(m)$ is changed to 0. This rotates the nominal $\mathbf{u}(2m+1) \in \mathcal{B}$ 90° counterclockwise to a point in $\mathcal{A} = 4\mathbf{Z}^2 + (1,1)$. Therefore, the modified late symbol becomes:

$$\mathbf{y}'(2m+1) = \mathbf{u}'(2m+1) + \mathbf{c}(2m+1) \in 4\mathbf{Z}^2 + (1,1) + (0,2) = \mathcal{B} \quad (10.40)$$

The resulting 4D symbol at the output of the prediction error filter is of the type $\mathcal{A} \times \mathcal{B}$ as before. ∎

10.6.1 Modifications Required in the Viterbi Decoder to Compensate for Superframe Bit Inversions

The superframe synchronization bit inversions do not change the sequence of states followed be the encoder. The only effect at the bit inversion times is to rotate the subset of the late symbol $\mathbf{y}(2m+1)$ at the output of the prediction error filter by 90° clockwise if the original 4D symbol has even parity ($Y_0(m) = 0$) or by 90° counterclockwise if the original 4D symbol has odd parity ($Y_0(m) = 1$). This effect can be compensated for in the Viterbi decoder at the synchronization times by modifying the 4D slicing subsets appropriately. When slicing $[\mathbf{y}(2m), \mathbf{y}'(2m+1)]$ for a state for which all the converging branches normally have even parity 4D subsets corresponding to $Y_0(m) = 0$, the late subset should be replaced by the nominal one rotated 90° clockwise. When the converging branches normally have odd parity corresponding to $Y_0(m) = 1$, the late subset should be replaced by the nominal one rotated 90° counterclockwise. Once the slicing to the modified subsets is performed, the decoder operates as usual.

EXAMPLE 10.2 Modifications for the 16-State Encoder

The 16-state convolutional encoder only uses $Y_1(m)$ and $Y_2(m)$ which correspond to $I_{i,j,1}$ and $z_0(m)$, respectively, in Figure 9.6 to drive encoder to the next state. Therefore, the trellis has four branches leaving each state and four entering each state. There are 16 4D subsets as show in Table 9.2. Eight of them have even parity and eight odd parity. To satisfy the property that all of the subsets with a given parity must be assigned to branches entering a state, pairs of subsets are assigned to each branch. One element of the pair is selected by $z_1(m) = 0$ which also corresponds to $Y_3(m)$, and the other by $z_1(m) = 1$. According to the mapping rules, changing z_1 from 0 to 1 adds a 180° rotation to both the early and late 2D symbols. A typical assignment of subsets to branches is shown in Figure 10.8 for even and odd parity 4D subsets. For example, $\mathcal{A} \times \mathcal{A}$ on the upper left-hand side of the figure must be paired paired with its 180° degree rotation $\mathcal{C} \times \mathcal{C}$. Notice that for each even parity 4D subset on the left of the figure there is an odd parity subset at the same level on the right side

differing only by a 90° clockwise rotation of the late 2D subset. These subsets have lattice partition variables that differ only in the parity selecting bit Y_0.

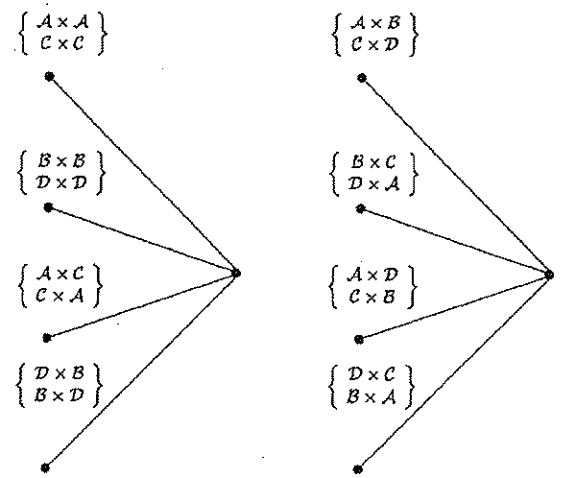


Figure 10.8. Typical 4D Subset Assignments to the Trellis Branches for the 16-State V.34 Code

The superframe synchronization bit inversions can be taken into account in the Viterbi decoder at the receiver when an inversion occurs by simply changing the branch 4D subsets to ones with the late subset rotated 90° clockwise if the original parity was even and by 90° counterclockwise if it was odd. This amounts to swapping the branch labels on the left and right of Figure 10.8 but leaving the states unchanged. Then the 4D symbol at the output of the prediction error filter can be sliced to these modified subsets and the cumulative metrics for the survivors to each state updated as usual. ∎

10.7 Receiver Operation

The block diagram of a simplified receiver is shown in Figure 10.9. The received signal is first passed through a passband adaptive equalizer with fractionally spaced taps. The equalizer output is demodulated to baseband resulting in the signal $\tilde{\mathbf{x}}(n)$ which is a noise corrupted version of the transmitted sequence

$x(n)$. This signal is passed through the prediction error filter which has the transfer function $H(z)$ given by (10.1). The output of the prediction error filter is

$$\bar{y}(n) = \bar{x}(n) + h_1\bar{x}(n-1) + h_2\bar{x}(n-2) + h_3\bar{x}(n-3) \tag{10.41}$$

The prediction error filter output is a noisy version of the trellis sequence $y(n)$ generated in the transmitter. The signal $\bar{y}(n)$ is applied to the Viterbi decoder to find the maximum likelihood estimate $\hat{y}(n)$ of the trellis sequence $y(n)$.

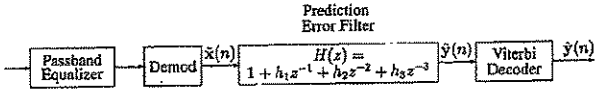


Figure 10.9. Receiver with Prediction Error Filter

The input $u(n)$ to the precoder in the transmitter can be reconstructed from the Viterbi decoder output. The block diagram of a system for performing the reconstruction is shown in Figure 10.10. In the transmitter, $Y(z)$ is formed from $X(z)$ by the filtering operation

$$Y(z) = H(z)X(z) \tag{10.42}$$

so $X(z)$ can be found from $Y(z)$ by the inverse filtering operation

$$X(z) = Y(z)/H(z) \tag{10.43}$$

This inverse operation is stable because it can be shown theoretically that the minimum mean-square error prediction error filter $H(z)$ has all its zeros inside the unit circle [59]. Using the standard single feedback loop transfer function equation, it follows that the transfer function from $\hat{y}(n)$ to $\hat{x}(n)$ in Figure 10.10 is $1/H(z)$. Therefore, when no decoding errors have occurred, $\hat{x}(n) = x(n)$. Then, $\hat{p}(n) = p(n)$ and with the same Modulo operation as in the transmitter $\hat{d}(n) = d(n)$. Since

$$x(n) = u(n) + d(n) \tag{10.44}$$

the input to the precoder in the transmitter can be reconstructed by the equation

$$\hat{u}(n) = \hat{x}(n) - \hat{d}(n) \tag{10.45}$$

when there are no decoding errors. The original input data bits can then be determined from the symbol sequence $\hat{u}(n)$ by an inverse of the MAP function.

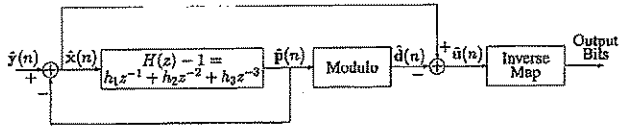


Figure 10.10. System to Compute the Transmitter's Precoder Input

Chapter 11

FAST EQUALIZER ADJUSTMENT BY USING A PERIODIC TRAINING SEQUENCE

The training sequence transmitted by a V.34 modem contains a 288 symbol segment called the *PP signal* which was designed to allow fast training of the equalizer in the receiver of the remote modem by using FFT techniques. This segment consists of six repetitions of a special 48 symbol sequence. Recommendation V.34 was the first V series recommendation to include a segment for fast equalizer training. Periodic sequences for equalizer training were used earlier in the first half of the 1970's by some proprietary modems like the Western Electric Bell 209 where it was called *cyclic equalization*. However, the Bell 209 used the slow LMS algorithm for equalizer adaptation. The theory and details of the V.34 PP signal are presented in Section 11.1. In Section 11.2, the transfer function of the optimal linear equalizer is derived in the case when an infinite number of taps is allowed. A method for fast calculation of the taps of a finite length equalizer by using the PP signal and FFT is described in Section 11.3.

11.1 The V.34 Periodic Training Sequence

The V.34 periodic equalizer training sequence contributed by Yuri Goldstein of General DataComm [31] is a special case of the CAZAC sequences discovered by A. Milewski [50]. The construction and properties of CAZAC sequences are derived in this section and the V.34 special case is given.

11.1.1 The Periodic Autocorrelation Function and CAZAC Sequences

Let $u_0, u_1, \ldots, u_{M-1}$ be a sequence of complex numbers which is repeated periodically to form the infinite sequence $x(n) = u_{\mathrm{mod}(n,M)}$. The sequence is said to have *constant amplitude* (CA) if $|x(n)|$ is the same constant for all n.

The periodic autocorrelation function for $x(n)$ is defined to be

$$\rho(n) = \sum_{m=0}^{M-1} x(n+m)\overline{x(m)} \tag{11.1}$$

The sequence $x(n)$ is said to have *zero autocorrelation function* (ZAC) if

$$\rho(n) = 0 \text{ for } n = 1, \ldots, M-1 \tag{11.2}$$

Sequences that are both constant amplitude and zero autocorrelation function are known as CAZAC sequences.

EXAMPLE 11.1 A 3-Point CAZAC Sequence

Consider the $M = 3$ point sequence $(u_0, u_1, u_2) = (1, e^{j2\pi/3}, 1)$. The first three values of the periodic autocorrelation function are

$$\begin{aligned}\rho(0) &= 1 \times 1 + e^{j2\pi/3}e^{-j2\pi/3} + 1 \times 1 = 3\\ \rho(1) &= e^{j2\pi/3} \times 1 + 1 \times e^{-j2\pi/3} + 1 \times 1 = 0\\ \rho(2) &= 1 \times 1 + 1 \times e^{-j2\pi/3} + e^{j2\pi/3} \times 1 = 0\end{aligned}$$

Therefore, the periodic extension of u_n is a CAZAC sequence with period 3. The V.34 48-point CAZAC sequence uses this sequence as a fundamental building block. ■

The discrete Fourier transform (DFT) of the M-point sequence u_n is

$$U_k = \sum_{n=0}^{M-1} u_n e^{-j2\pi nk/M} \text{ for } k = 0, \ldots, M-1 \tag{11.3}$$

It can be shown [59] that the DFT of the periodic autocorrelation function of an M-point sequence is

$$R_k = \sum_{n=0}^{M-1} \rho(n)e^{-j2\pi nk/M} = |U_k|^2 \text{ for } k = 0, \ldots, M-1 \tag{11.4}$$

If u_n is a CAZAC sequence, then $\rho(0)$ is some constant c and $\rho(n) = 0$ for $n = 1, \ldots, M-1$ and, according to (11.4)

$$R_k = |U_k|^2 = c \text{ for all } k \tag{11.5}$$

so the spectrum of u_n is flat.

Another DFT relationship is

$$\frac{1}{M}\sum_{\ell=0}^{M-1} U_{\text{mod}(\ell+k,M)}\overline{U}_\ell = \sum_{n=0}^{M-1} |u_n|^2 e^{-j2\pi nk/M} \tag{11.6}$$

The left-hand side of this equation is the periodic autocorrelation function for the DFT sequence U_k. Now suppose u_n is a constant amplitude sequence, so that $|u_n|^2 = \alpha$ for all n. Substituting this CA property into the right-hand side of (11.6) gives

$$\frac{1}{M}\sum_{\ell=0}^{M-1} U_{\text{mod}(\ell+k,M)}\overline{U}_\ell = \begin{cases} M\alpha & \text{for } k = 0\\ 0 & \text{for } k = 1, \ldots, M-1\end{cases} \tag{11.7}$$

Therefore, the DFT of a CAZAC sequence is also a CAZAC sequence.

11.1.2 Constructing a CAZAC Sequence of Length MK^2 from one of Length M

Milewski [50] shows how to construct CAZAC sequences of length $N = MK^2$ from a CAZAC sequence of length M. First, consider the $L = MK$ point sequence

$$x_n = u_{\text{mod}(n,M)} \text{ for } n = 0, \ldots MK-1 \tag{11.8}$$

where u_n is a CAZAC sequence with period M. The sequence x_n consists of K repetitions of the basic sequence u_n. It is easy to show that the periodic extension of the $L = MK$ point sequence x_n is also a CAZAC sequence with period M.

A useful observation in the proof of Milewski's construction is that the L-point DFT of x_n has the form

$$\begin{aligned}[X_0, \ldots, X_{L-1}] = [&KU_0, \underbrace{0, \ldots, 0}_{K-1}, KU_1, \underbrace{0, \ldots, 0}_{K-1},\\ &\ldots, KU_{M-1}, \underbrace{0, \ldots, 0}_{K-1}]\end{aligned} \tag{11.9}$$

This will now be shown. The $L = MK$ point DFT is

$$X_k = \sum_{n=0}^{L-1} x_n e^{-j2\pi nk/L} \text{ for } k = 0, \ldots, L-1 \tag{11.10}$$

Let $n = \alpha M + \beta$ for $\alpha = 0, \ldots, K-1$ and $\beta = 0, \ldots M-1$. Then, the sum can be expressed as

$$\begin{aligned}X_k &= \sum_{\alpha=0}^{K-1}\sum_{\beta=0}^{M-1} x_{\alpha M+\beta}e^{-j2\pi(\alpha M+\beta)k/L}\\ &= \sum_{\alpha=0}^{K-1}\left[\sum_{\beta=0}^{M-1} u_\beta e^{-j2\pi\beta k/L}\right]e^{-j2\pi\alpha Mk/L}\end{aligned}$$

$$= \sum_{\beta=0}^{M-1} u_\beta e^{-j\frac{2\pi}{MK}\beta k} \sum_{\alpha=0}^{K-1} e^{-j\frac{2\pi}{K}\alpha k} \text{ for } k = 0, \ldots, L-1 \quad (11.11)$$

The last sum on the right-hand side has the values

$$\sum_{\alpha=0}^{K-1} e^{-j\frac{2\pi}{K}\alpha k} = \begin{cases} K & \text{when } k \text{ is a multiple of } K \\ 0 & \text{otherwise} \end{cases} \quad (11.12)$$

Also,

$$\sum_{\beta=0}^{M-1} u_\beta e^{-j\frac{2\pi}{MK}\beta k} = U_\ell \text{ for } k = \ell K \quad (11.13)$$

Therefore,

$$X_k = \begin{cases} KU_\ell & \text{for } k = \ell K \\ 0 & \text{otherwise} \end{cases} \quad (11.14)$$

The next step in the construction is to form a matrix $\mathbf{Y}$ which has $L = MK$ rows and K columns with the elements

$$\mathbf{Y}_{n,q} = x_n e^{j2\pi nq/L} \text{ for } n = 0, \ldots, L-1 \text{ and } q = 0, \ldots, K-1 \quad (11.15)$$

Notice that the q-th column of this matrix is the L-point sequence x_n multiplied by the complex exponential $e^{j2\pi nq/L}$ and according to the frequency shift theorem

$$\text{DFT}_L\{\mathbf{Y}_{n,q}\} = X_{k-q} \text{ for } q = 0, \ldots, K-1 \quad (11.16)$$

Thus, the DFT of the q-th column corresponds to the sequence in (11.9) cyclically shifted right q positions.

One period of Milewski's MK^2 point CAZAC sequence is formed by concatenating the rows of $\mathbf{Y}$. Let the desired CAZAC sequence be called $s(n)$. The time index n can be expressed as

$$n = \ell K + r \text{ for } 0 \le n \le MK^2 - 1 \quad (11.17)$$

where

$$\ell = \text{int}(n/K) \text{ and } r = \text{mod}(n, K) \quad (11.18)$$

The function $\text{int}(x) = \lfloor x \rfloor$ is the integer part of x and the function $\text{mod}(n, K)$ is the remainder when n is divided by K. The ranges for the integer ℓ and r are

$$0 \le \ell \le MK - 1 \text{ and } 0 \le r \le K - 1$$

Then

$$s(n) = \mathbf{Y}_{\ell,r} \text{ for } n = 0, \ldots, MK^2 - 1 \quad (11.19)$$

11.1.2.1 Proof that $s(n)$ is a CAZAC Sequence

We will use the following version of Parseval's theorem for DFT's in proving that $s(n)$ is a CAZAC sequence. Let a_n and b_n be N-point sequences with DFT's A_k and B_k. Then Parseval's theorem is

$$\sum_{n=0}^{N-1} a_n \overline{b}_n = \frac{1}{N} \sum_{k=0}^{N-1} A_k \overline{B}_k \quad (11.20)$$

By using Parseval's theorem, it can be shown that the columns of $\mathbf{Y}$ are orthogonal. That is

$$\sum_{n=0}^{L-1} \mathbf{Y}_{n,q} \overline{\mathbf{Y}}_{n,q'} = \frac{1}{L} \sum_{k=0}^{L-1} X_{k-q} \overline{X}_{k-q'} = 0 \text{ for } q \ne q' \quad (11.21)$$

since the nonzero positions in X_{k-q} and $X_{k-q'}$ do not overlap.

The autocorrelation function for $s(n)$ is

$$\rho(n) = \sum_{m=0}^{N-1} s(n+m)\overline{s}(m) \quad (11.22)$$

Now let

$$m = \alpha K + \beta \text{ for } \alpha = 0, \ldots, MK-1 \quad \beta = 0, \ldots, K-1 \quad (11.23)$$

Then

$$\rho(n) = \sum_{\beta=0}^{K-1} \sum_{\alpha=0}^{L-1} s(n + \alpha K + \beta) \overline{Y}_{\alpha,\beta} \quad (11.24)$$

Also let

$$n + \beta = \alpha' K + \beta' \quad (11.25)$$

Then, the autocorrelation function can be written as

$$\begin{aligned} \rho(n) &= \sum_{\beta=0}^{K-1} \sum_{\alpha=0}^{L-1} s\left((\alpha + \alpha')K + \beta'\right) \overline{\mathbf{Y}}_{\alpha,\beta} \\ &= \sum_{\beta=0}^{K-1} \sum_{\alpha=0}^{L-1} \mathbf{Y}_{\alpha+\alpha',\beta'} \overline{\mathbf{Y}}_{\alpha,\beta} \end{aligned} \quad (11.26)$$

The inner summation over α is the inner product of column β of $\mathbf{Y}$ and column β' of $\mathbf{Y}$ cyclically shifted by α' positions.

First consider the case where n is not a multiple of K. It follows from (11.25) that $\beta \ne \beta'$ in this case. The DFT of the shifted column of $\mathbf{Y}$ is

$$\text{DFT}_L\{\mathbf{Y}_{n+\alpha',\beta'}\} = \text{DFT}_L\{\mathbf{Y}_{n,\beta'}\} e^{j2\pi k\alpha'/L} = X_{k-\beta'} e^{j2\pi k\alpha'/L} \quad (11.27)$$

Also, according to (11.16) the DFT of the unshifted version of column β' is

$$\text{DFT}\{\mathbf{Y}_{n,\beta'}\} = X_{k-\beta'} \tag{11.28}$$

Therefore, the nonzero positions in the DFT's of the shifted and unshifted versions of column β' are at the same positions. Using Parseval's theorem and the fact that $\beta \neq \beta'$ gives

$$\sum_{\alpha=0}^{L-1} \mathbf{Y}_{\alpha+\alpha',\beta'} \overline{\mathbf{Y}}_{\alpha,\beta} = \frac{1}{L} \sum_{k=0}^{L-1} X_{k-\beta'} e^{j2\pi k\alpha'/L} \overline{X}_{k-\beta} = 0 \tag{11.29}$$

So, from (11.26) we find that $\rho(n) = 0$ for this case.

Now let n be a multiple of K which implies that $\beta = \beta'$ and $n = \alpha' K$. With these conditions, the autocorrelation function can be expressed as

$$\begin{aligned}
\rho(\alpha' K) &= \sum_{\beta=0}^{K-1} \sum_{\alpha=0}^{L-1} \mathbf{Y}_{\alpha+\alpha',\beta} \overline{\mathbf{Y}}_{\alpha,\beta} \\
&= \sum_{\beta=0}^{K-1} \sum_{\alpha=0}^{L-1} u_{\text{mod}(\alpha+\alpha',M)} e^{j\frac{2\pi}{L}\beta(\alpha+\alpha')} \overline{u}_{\text{mod}(\alpha,M)} e^{-j\frac{2\pi}{L}\beta\alpha} \\
&= \sum_{\beta=0}^{K-1} \sum_{\alpha=0}^{L-1} u_{\text{mod}(\alpha+\alpha',M)} \overline{u}_{\text{mod}(\alpha,M)} e^{-j\frac{2\pi}{L}\beta\alpha'} \\
&= \left(\sum_{\alpha=0}^{L-1} u_{\text{mod}(\alpha+\alpha',M)} \overline{u}_{\text{mod}(\alpha,M)} \right) \left(\sum_{\beta=0}^{K-1} e^{-j\frac{2\pi}{L}\beta\alpha'} \right)
\end{aligned} \tag{11.30}$$

There are two cases to consider in this last equation: (1) α' is a multiple of M and (2) α' is not a multiple of M. The results are:

Case 1
Let $\alpha' = pM$ for some integer p. Also assume α' is not a multiple of K since when it is a multiple of K, then $n = \alpha' K = cMK^2$ which is a multiple of the sequence period N. This also implies p is not a multiple of K. Then (11.30) becomes

$$\rho(\alpha' K) = \left(\sum_{\alpha=0}^{L-1} |u_{\text{mod}(\alpha,M)}|^2 \right) \left(\sum_{\beta=0}^{K-1} e^{-j\frac{2\pi}{K}\beta p} \right) \tag{11.31}$$

Since p is not a multiple of K, the far right-hand sum is zero.

Case 2
Now assume α' is not a multiple of M. Then the first factor on the right-hand side of (11.30) is zero because u_n is a CAZAC sequence.

11.1.3 The V.34 CAZAC Sequence

The following quotation is the definition of the PP sequence in Recommendation V.34 [34]:

10.1.3.6 PP

Signal PP consists of six periods of a 48-symbol sequence and is used by the remote modem for training its equalizer. PP(i), $i = 0, 1, \ldots, 287$, is defined as follows:

Set $i = 4k + \ell$ where
$k = 0, 1, 2, \ldots, 71$ and
$\ell = 0, 1, 2, 3$ for each k.
Then:

$$\begin{aligned}
\text{PP}(i) &= e^{j\pi(k\ell+4)/6} \quad \text{, if } k \text{ modulo } 3 = 1 \\
&= e^{j\pi k\ell/6} \quad \text{, otherwise}
\end{aligned} \qquad \text{Equation 10-1/V.34}$$

PP(0) is transmitted first.

The entire 288-point PP sequence consists of six repetitions of its first 48 points PP(0), ..., PP(47). This 48-point sequence is a CAZAC sequence with the parameters $M = 3$, $K = 4$, $L = MK = 12$, and $N = LK = MK^2 = 48$. The points have unity magnitude and one of 12 angles equally spaced between 0 and 360 degrees. The matrix $\mathbf{Y}$ has 12 rows and 4 columns in this case. The first column of $\mathbf{Y}$ consists of $K = 4$ repetitions of the $M = 3$ point CAZAC sequence $(u_0, u_1, u_2) = (1, e^{j2\pi/3}, 1)$ discussed in Example 11.1.

The row of $\mathbf{Y}$ is selected by $\text{mod}(k, 12)$ using the index k in the V.34 definition. The six repetitions include $12 \times 6 = 72$ rows so k ranges from 0 to 71. For $\text{mod}(k, 3) = 0$ or 2 the elementary sequence elements $u_0 = u_2 = 1$ are used. For $\text{mod}(k, 3) = 1$, the element $u_1 = e^{j2\pi/3}$ is used.

The index ℓ specifies the column of $\mathbf{Y}$. The V.34 definition can be rewritten as

$$\text{PP}(i) = \begin{cases} 1 \times e^{j2\pi k\ell/12} & \text{for } \text{mod}(k,3) = 0 \\ e^{j2\pi/3} \times e^{j2\pi k\ell/12} & \text{for } \text{mod}(k,3) = 1 \\ 1 \times e^{j2\pi k\ell/12} & \text{for } \text{mod}(k,3) = 2 \end{cases} \tag{11.32}$$

This explicitly shows how the V.34 definition uses the fundamental 3-point sequence u_n and multiplies the columns by the complex exponentials specified by (11.15).

11.2 The Optimal Fractionally Spaced Equalizer

The PP sequence was included in Recommendation V.34 to allow fast equalizer training at the far end receiver. The frequency response of the optimum equalizer is derived in this section and a method of using the PP sequence to approximate this response is presented in the following section. The block diagram for a discrete-time model of a QAM system with a fractionally spaced

equalizer is shown in Figure 11.1. We will assume that QAM symbols are sent every T seconds or at the symbol rate of $f_s = 1/T$ Hz or $\omega_s = 2\pi/T$ radians/second.

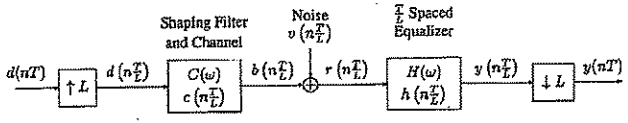


Figure 11.1. QAM System with a Fractionally Spaced Equalizer

Suppose the baseband sequence of QAM constellation points is $a(nT)$. The signal $d(nT)$ represents the rotated QAM symbol sequence $a(nT)e^{j\omega_c nT}$ where ω_c is the carrier frequency. We will assume $d(nT)$ is a zero-mean uncorrelated sequence with variance

$$\mathrm{E}\{|d(nT)|^2\} = \sigma_d^2 \tag{11.33}$$

In practice, $d(nT)$ is impulse modulated and applied to a continuous-time passband shaping filter to form the transmitted signal which is applied to the telephone line channel. If the combined impulse response of the shaping filter and channel is $c(t)$, the noiseless channel output is

$$b(t) = \sum_{k=-\infty}^{\infty} d(kT)c(t - kT) \tag{11.34}$$

The bandwidth of the V.34 transmitted QAM signal is about 12% greater than the symbol rate f_s and its spectrum is centered about a carrier frequency in the center of the voiceband telephone channel passband. Therefore, the combined transmitter shaping filter and channel can be modeled by the discrete-time filter with impulse response $c\left(n\frac{T}{L}\right)$ shown in Figure 11.1 which operates at a sampling rate L times faster than the symbol rate to prevent aliasing. Typically, a value of $L = 3$ is sufficient. The combined shaping filter and channel frequency response for this discrete-time model is

$$C(\omega) = \sum_{n=-\infty}^{\infty} c\left(n\tfrac{T}{L}\right) e^{-j\omega n\frac{T}{L}} \tag{11.35}$$

The block containing $\uparrow L$ creates the fictitious fast sampled signal

$$d\left(n\tfrac{T}{L}\right) = \begin{cases} d(kT) & \text{for } n = kL \\ 0 & \text{elsewhere} \end{cases} \tag{11.36}$$

This is the original sequence with $L-1$ zeros inserted between each original sample. The noiseless signal samples at the channel output are

$$b\left(n\tfrac{T}{L}\right) = \sum_{k=-\infty}^{\infty} d(kT)c\left(n\tfrac{T}{L} - kT\right) \tag{11.37}$$

The signal $v\left(n\frac{T}{L}\right)$ represents a zero-mean, complex, stationary noise sequence that is independent of the data signal and has the autocorrelation function

$$R_{vv}(m) = \mathrm{E}\left\{v\left((n+m)\tfrac{T}{L}\right)\overline{v\left(n\tfrac{T}{L}\right)}\right\} \tag{11.38}$$

and power spectral density

$$S_{vv}(\omega) = \sum_{m=-\infty}^{\infty} R_{vv}(m)e^{-j\omega m\frac{T}{L}} \tag{11.39}$$

The received signal is

$$r\left(n\tfrac{T}{L}\right) = b\left(n\tfrac{T}{L}\right) + v\left(n\tfrac{T}{L}\right) \tag{11.40}$$

This received signal is passed through a linear equalizer with T/L spaced taps, $h\left(n\frac{T}{L}\right)$, and transfer function

$$H(\omega) = \sum_{n=-\infty}^{\infty} h\left(n\tfrac{T}{L}\right) e^{-j\omega n\frac{T}{L}} \tag{11.41}$$

The equalizer output is down-sampled by a factor of L to get the estimate, $y(nT)$, of the transmitted passband symbol $d(nT)$.

It will be shown in the following subsection that the optimum equalizer transfer function is

$$H(\omega) = \frac{\overline{C(\omega)}}{S_{vv}(\omega)} \frac{\sigma_d^2}{\sigma_d^2 \frac{1}{L}\sum_{m=0}^{L-1} \frac{|C(\omega - m\omega_s)|^2}{S_{vv}(\omega - m\omega_s)} + 1} \tag{11.42}$$

and in subsection (11.2.2) that the corresponding mean-squared error is

$$\Lambda = \sigma_d^2 \frac{1}{\omega_s} \int_{-\frac{\omega_s}{2}}^{\frac{\omega_s}{2}} \frac{1}{\sigma_d^2 \frac{1}{L}\sum_{m=0}^{L-1} \frac{|C(\omega - m\omega_s)|^2}{S_{vv}(\omega - m\omega_s)} + 1} \, d\omega \tag{11.43}$$

EXAMPLE 11.2 High Signal-to-Noise Ratio Approximation

When the noise power is very small relative to the signal power, the 1 can be dropped in the denominator and the optimum equalizer transfer function is closely approximated by

$$H(\omega) = \frac{\overline{C(\omega)}}{S_{vv}(\omega)} \frac{1}{\frac{1}{L}\sum_{m=0}^{L-1} \frac{|C(\omega - m\omega_s)|^2}{S_{vv}(\omega - m\omega_s)}} \tag{11.44}$$

The overall transfer function of the equalizer cascaded with the channel is

$$Q(\omega) = C(\omega)H(\omega) = \frac{|C(\omega)|^2}{S_{vv}(\omega)} \frac{1}{\frac{1}{L}\sum_{m=0}^{L-1} \frac{|C(\omega - m\omega_s)|^2}{S_{vv}(\omega - m\omega_s)}} \tag{11.45}$$

Notice that the denominator of the second factor on the right-hand side is the down-sampling by a factor of L aliasing formula applied to the first factor. When the sequence corresponding to $Q(\omega)$ is down-sampled by a factor of L, the resulting sequence has the transform

$$\begin{aligned} Q^*(\omega) &= \frac{1}{L}\sum_{k=0}^{L-1} Q(\omega - k\omega_s) \\ &= \frac{1}{L}\sum_{k=0}^{L-1} \frac{|C(\omega - k\omega_s)|^2}{S_{vv}(\omega - k\omega_s)} \frac{1}{\frac{1}{L}\sum_{m=0}^{L-1} \frac{|C(\omega - k\omega_s - m\omega_s)|^2}{S_{vv}(\omega - k\omega_s - m\omega_s)}} \end{aligned} \tag{11.46}$$

But the sum over m in the denominator has period ω_s, so

$$Q^*(\omega) = \frac{1}{L}\sum_{k=0}^{L-1} \frac{|C(\omega - k\omega_s)|^2}{S_{vv}(\omega - k\omega_s)} \frac{1}{\frac{1}{L}\sum_{m=0}^{L-1} \frac{|C(\omega - m\omega_s)|^2}{S_{vv}(\omega - m\omega_s)}} = 1 \tag{11.47}$$

Therefore, the symbol rate samples of the combined response are

$$q(nT) = \begin{cases} 1 & \text{for } n = 0 \\ 0 & \text{otherwise} \end{cases} \tag{11.48}$$

so there is no intersymbol interference and $H(\omega)$ perfectly equalizes the channel. ∎

EXAMPLE 11.3 White Noise Case

When the noise spectrum is flat, that is, $S_{vv}(\omega) = \sigma_v^2$ the optimum equalizer is

$$H(\omega) = \frac{\overline{C(\omega)}}{\frac{1}{L}\sum_{m=0}^{L-1} |C(\omega - m\omega_s)|^2 + \frac{\sigma_v^2}{\sigma_d^2}} \tag{11.49}$$

with the MSE

$$\Lambda = \sigma_v^2 \frac{1}{\omega_s} \int_{-\frac{\omega_s}{2}}^{\frac{\omega_s}{2}} \frac{1}{\frac{1}{L}\sum_{m=0}^{L-1} |C(\omega - m\omega_s)|^2 + \frac{\sigma_v^2}{\sigma_d^2}} d\omega \tag{11.50}$$

When there is no noise, that is, $\sigma_v^2 = 0$, the optimum equalizer reduces to

$$H(\omega) = \frac{\overline{C(\omega)}}{\frac{1}{L}\sum_{m=0}^{L-1} |C(\omega - m\omega_s)|^2} \tag{11.51}$$

and the MSE is zero. This equalizer follows from the previous example with $S_{vv}(\omega) = \sigma_v^2$ and also perfectly equalizes the channel. ∎

11.2.1 Derivation of the Optimum Linear Equalizer

The equalizer input and down-sampled output are related by the equation

$$\begin{aligned} y(nT) &= \sum_{i=-\infty}^{\infty} h\left(i\tfrac{T}{L}\right) r\left(nT - i\tfrac{T}{L}\right) \\ &= \sum_{i=-\infty}^{\infty} h\left(i\tfrac{T}{L}\right) b\left(nT - i\tfrac{T}{L}\right) \\ &\quad + \sum_{i=-\infty}^{\infty} h\left(i\tfrac{T}{L}\right) v\left(nT - i\tfrac{T}{L}\right) \end{aligned} \tag{11.52}$$

On using (11.37) to evaluate $b\left(nT - i\frac{T}{L}\right)$, this becomes

$$\begin{aligned} y(nT) &= \left[\sum_{k=-\infty}^{\infty} d(kT) \sum_{i=-\infty}^{\infty} h\left(i\tfrac{T}{L}\right) c\left(nT - i\tfrac{T}{L} - kT\right)\right] \\ &\quad + \sum_{i=-\infty}^{\infty} h\left(i\tfrac{T}{L}\right) v\left(nT - i\tfrac{T}{L}\right) \end{aligned} \tag{11.53}$$

The instantaneous error at the equalizer output is

$$e(nT) = y(nT) - d(nT) \tag{11.54}$$

The problem is to find the equalizer impulse response $h\left(n\frac{T}{L}\right)$ or transfer function $H(\omega)$ that minimizes the mean-squared error

$$\Lambda = \mathrm{E}\left\{|e(nT)|^2\right\} \tag{11.55}$$

The method that will be used parallels the approach taken in the classic book by Luck, Salz, and Weldon [49] for continuous time systems. The equalizer output is a weighted sum of the observed signal $r\left(nT - i\frac{T}{L}\right)$ over all i. According to the orthogonality principle [59], the mean-squared error is minimized by making the error orthogonal to the observations. Therefore, the optimum equalizer must satisfy the equation

$$\mathrm{E}\left\{[y(nT) - d(nT)]\overline{r\left(nT - i\frac{T}{L}\right)}\right\} = 0 \text{ for all } i \tag{11.56}$$

or

$$\mathrm{E}\left\{y(nT)\overline{r\left(nT - i\frac{T}{L}\right)}\right\} = \mathrm{E}\left\{d(nT)\overline{r\left(nT - i\frac{T}{L}\right)}\right\} \text{ for all } i \tag{11.57}$$

First, the left-hand side of (11.57) will be evaluated. Using (11.53) for $y(nT)$ in (11.57) gives

$$\begin{aligned}
&\mathrm{E}\left\{y(nT)\overline{r\left(nT - i\frac{T}{L}\right)}\right\} \\
&\quad = \mathrm{E}\left\{\sum_{k=-\infty}^{\infty} d(kT) \sum_{\ell=-\infty}^{\infty} h\left(\ell\frac{T}{L}\right) c\left(nT - \ell\frac{T}{L} - kT\right) \overline{r\left(nT - i\frac{T}{L}\right)}\right. \\
&\qquad \left. + \sum_{\ell=-\infty}^{\infty} h\left(\ell\frac{T}{L}\right) v\left(nT - \ell\frac{T}{L}\right) \overline{r\left(nT - i\frac{T}{L}\right)}\right\}
\end{aligned} \tag{11.58}$$

It follows from (11.37) and (11.40) that

$$r\left(nT - i\frac{T}{L}\right) = \sum_{p=-\infty}^{\infty} d(pT) c\left(nT - i\frac{T}{L} - pT\right) + v\left(nT - i\frac{T}{L}\right) \tag{11.59}$$

Substituting this last result into (11.58), interchanging expectation and summation, and using the fact that the data signal and noise are uncorrelated yields

$$\begin{aligned}
&\mathrm{E}\left\{y(nT)\overline{r\left(nT - i\frac{T}{L}\right)}\right\} \\
&\quad = \left[\sum_{k=-\infty}^{\infty} \sum_{p=-\infty}^{\infty} \mathrm{E}\left\{d(kT)\overline{d(pT)}\right\} \overline{c\left(nT - i\frac{T}{L} - pT\right)} \times \right.
\end{aligned}$$

$$\left. \sum_{\ell=-\infty}^{\infty} h\left(\ell\frac{T}{L}\right) c\left(nT - \ell\frac{T}{L} - kT\right)\right]$$
$$+ \sum_{\ell=-\infty}^{\infty} h\left(\ell\frac{T}{L}\right) \mathrm{E}\left\{v\left(nT - \ell\frac{T}{L}\right) \overline{v\left(nT - i\frac{T}{L}\right)}\right\} \tag{11.60}$$

Using the fact that $\mathrm{E}\{d(nT)\overline{d(pT)}\} = 0$ for $n \neq 0$ and the definition of the noise autocorrelation function, this reduces to

$$\begin{aligned}
&\mathrm{E}\left\{y(nT)\overline{r\left(nT - i\frac{T}{L}\right)}\right\} \\
&\quad = \left[\sigma_d^2 \sum_{k=-\infty}^{\infty} \overline{c\left(nT - i\frac{T}{L} - kT\right)} \sum_{\ell=-\infty}^{\infty} h\left(\ell\frac{T}{L}\right) c\left(nT - \ell\frac{T}{L} - kT\right)\right] \\
&\qquad + \sum_{\ell=-\infty}^{\infty} h\left(\ell\frac{T}{L}\right) R_{vv}\left(i\frac{T}{L} - \ell\frac{T}{L}\right)
\end{aligned} \tag{11.61}$$

Finally, changing the summation index on the right-hand side to replace $n - k$ by k gives

$$\begin{aligned}
&\mathrm{E}\left\{y(nT)\overline{r\left(nT - i\frac{T}{L}\right)}\right\} \\
&\quad = \left[\sigma_d^2 \sum_{k=-\infty}^{\infty} \overline{c\left(kT - i\frac{T}{L}\right)} \sum_{\ell=-\infty}^{\infty} h\left(\ell\frac{T}{L}\right) c\left(kT - \ell\frac{T}{L}\right)\right] \\
&\qquad + \sum_{\ell=-\infty}^{\infty} h\left(\ell\frac{T}{L}\right) R_{vv}\left(i\frac{T}{L} - \ell\frac{T}{L}\right)
\end{aligned} \tag{11.62}$$

Now the right-hand side of (11.57) will be evaluated. Substituting (11.59) into the right-hand side of (11.57) yields

$$\mathrm{E}\left\{d(nT)\overline{r\left(nT - i\frac{T}{L}\right)}\right\} = \mathrm{E}\left\{d(nT) \sum_{p=-\infty}^{\infty} \overline{d(pT) c\left(nT - i\frac{T}{L} - pT\right)}\right\} \tag{11.63}$$

Interchanging expectation and summation and using the fact that the data sequence is uncorrelated give

$$\begin{aligned}
\mathrm{E}\left\{d(nT)\overline{r\left(nT - i\frac{T}{L}\right)}\right\} &= \sum_{p=-\infty}^{\infty} \mathrm{E}\left\{d(nT)\overline{d(pT)}\right\} \overline{c\left(nT - i\frac{T}{L} - pT\right)} \\
&= \sigma_d^2 \overline{c\left(-i\frac{T}{L}\right)}
\end{aligned} \tag{11.64}$$

Putting (11.62) and (11.64) together, we see that the optimum equalizer must satisfy the following equation for all i:

$$\left[\sigma_d^2 \sum_{k=-\infty}^{\infty} \overline{c\left(kT - i\frac{T}{L}\right)} \sum_{\ell=-\infty}^{\infty} h\left(\ell\frac{T}{L}\right) c\left(kT - \ell\frac{T}{L}\right)\right] + \sum_{\ell=-\infty}^{\infty} h\left(\ell\frac{T}{L}\right) R_{vv}\left(i\frac{T}{L} - \ell\frac{T}{L}\right) = \sigma_d^2 \overline{c\left(-i\frac{T}{L}\right)} \tag{11.65}$$

As the first step in solving (11.65) for $h\left(i\frac{T}{L}\right)$, let

$$\sum_{\ell=-\infty}^{\infty} h\left(\ell\frac{T}{L}\right) c\left(kT - \ell\frac{T}{L}\right) = z_k \tag{11.66}$$

Then (11.65) can be written as

$$\sigma_d^2 \sum_{k=-\infty}^{\infty} \overline{c\left(kT - i\frac{T}{L}\right)} z_k + \sum_{\ell=-\infty}^{\infty} h\left(\ell\frac{T}{L}\right) R_{vv}\left(i\frac{T}{L} - \ell\frac{T}{L}\right) = \sigma_d^2 \overline{c\left(-i\frac{T}{L}\right)} \text{ for all } i \tag{11.67}$$

Taking the discrete-time Fourier transform of (11.67) with respect to i, that is, multiplying by $e^{-j\omega i\frac{T}{L}}$ and summing over all i gives

$$\sigma_d^2 \overline{C(\omega)} \sum_{k=-\infty}^{\infty} z_k e^{-j\omega kT} + H(\omega) S_{vv}(\omega) = \sigma_d^2 \overline{C(\omega)} \tag{11.68}$$

Solving for $H(\omega)$ gives

$$H(\omega) = \sigma_d^2 \left[\sum_{k=-\infty}^{\infty} z_k e^{-j\omega kT} - 1\right] \frac{\overline{C(\omega)}}{S_{vv}(\omega)} \tag{11.69}$$

Let

$$G(\omega) = \frac{\overline{C(\omega)}}{S_{vv}(\omega)} = \sum_{i=-\infty}^{\infty} g\left(i\frac{T}{L}\right) e^{-j\omega i\frac{T}{L}} \tag{11.70}$$

Then $h\left(i\frac{T}{L}\right)$ must have the form

$$h\left(i\frac{T}{L}\right) = \sum_{n=-\infty}^{\infty} \alpha_n g\left(i\frac{T}{L} - nT\right) \tag{11.71}$$

for some set of constants $\{\alpha_n\}$. The discrete-time Fourier transform of $h\left(i\frac{T}{L}\right)$ with respect to i is

$$H(\omega) = \sum_{i=-\infty}^{\infty} h\left(i\frac{T}{L}\right) e^{-j\omega i\frac{T}{L}} = G(\omega) A(\omega) \tag{11.72}$$

where

$$A(\omega) = \sum_{n=-\infty}^{\infty} \alpha_n e^{-j\omega nT} \tag{11.73}$$

Substituting (11.71) into (11.66) yields

$$z_k = \sum_{n=-\infty}^{\infty} \alpha_n \sum_{\ell=-\infty}^{\infty} g\left(\ell\frac{T}{L}\right) c\left(kT - nT - \ell\frac{T}{L}\right) \tag{11.74}$$

Then, substituting this formula and (11.71) into (11.67) gives

$$\sigma_d^2 \sum_{k=-\infty}^{\infty} \overline{c\left(kT - i\frac{T}{L}\right)} \sum_{n=-\infty}^{\infty} \alpha_n \sum_{\ell=-\infty}^{\infty} g\left(\ell\frac{T}{L}\right) c\left(kT - nT - \ell\frac{T}{L}\right) + \sum_{n=-\infty}^{\infty} \alpha_n \sum_{\ell=-\infty}^{\infty} g\left(\ell\frac{T}{L} - nT\right) R_{vv}\left(i\frac{T}{L} - \ell\frac{T}{L}\right) = \sigma_d^2 \overline{c\left(-i\frac{T}{L}\right)} \tag{11.75}$$

which has the discrete-time Fourier transform with respect to i

$$\sigma_d^2 \sum_{k=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} \alpha_n e^{-j\omega kT} \overline{C(\omega)} \sum_{\ell=-\infty}^{\infty} g\left(\ell\frac{T}{L}\right) c\left(kT - nT - \ell\frac{T}{L}\right) + \sum_{n=-\infty}^{\infty} \alpha_n G(\omega) e^{-j\omega nT} S_{vv}(\omega) = \sigma_d^2 \overline{C(\omega)} \tag{11.76}$$

Remember that $G(\omega) S_{vv}(\omega) = \overline{C(\omega)}$ so $\overline{C(\omega)}$ can be cancelled from each term resulting in

$$\sigma_d^2 \sum_{n=-\infty}^{\infty} \alpha_n \left[\sum_{k=-\infty}^{\infty} e^{-j\omega kT} \sum_{\ell=-\infty}^{\infty} g\left(\ell\frac{T}{L}\right) c\left(kT - nT - \ell\frac{T}{L}\right)\right] + \sum_{n=-\infty}^{\infty} \alpha_n e^{-j\omega nT} = \sigma_d^2 \tag{11.77}$$

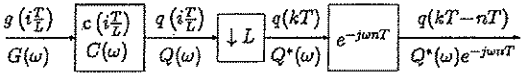


Figure 11.2. System the Generates the Sum in the Rectangular Brackets

A system for generating the sum over ℓ in the rectangular brackets in (11.77) is shown in Figure 11.2. The system input $g\left(i\frac{T}{L}\right)$ passes through the filter

$c\left(i\frac{T}{L}\right)$ to cause the output

$$q\left(i\tfrac{T}{L}\right) = \sum_{\ell=-\infty}^{\infty} g\left(\ell\tfrac{T}{L}\right) c\left(i\tfrac{T}{L} - \ell\tfrac{T}{L}\right) \tag{11.78}$$

which has the discrete-time Fourier transform with respect to i

$$Q(\omega) = G(\omega)C(\omega) \tag{11.79}$$

This signal is down-sampled by a factor of L resulting in the signal

$$q(kT) = \sum_{\ell=-\infty}^{\infty} g\left(\ell\tfrac{T}{L}\right) c\left(kT - \ell\tfrac{T}{L}\right) \tag{11.80}$$

According to the skip sampling or down-sampling aliasing formula [59], the transform of $q(kT)$ with respect to k is

$$\begin{aligned} Q^*(\omega) &= \sum_{k=-\infty}^{\infty} q(kT)e^{-j\omega kT} = \frac{1}{L}\sum_{m=0}^{L-1} Q(\omega - m\omega_s) \\ &= \frac{1}{L}\sum_{m=0}^{L-1} G(\omega - m\omega_s)C(\omega - m\omega_s) \end{aligned} \tag{11.81}$$

where $\omega_s = 2\pi/T$ is the symbol rate. The output of the final block containing $e^{-j\omega kT}$ is the delayed signal

$$q(kT - nT) = \sum_{\ell=-\infty}^{\infty} g\left(\ell\tfrac{T}{L}\right) c\left(kT - nT - \ell\tfrac{T}{L}\right) \tag{11.82}$$

which is just the desired sum and has the transform with respect to k

$$Q^*(\omega)e^{-j\omega nT} = \frac{1}{L}\sum_{m=0}^{L-1} G(\omega - m\omega_s)C(\omega - m\omega_s)e^{-j\omega nT} \tag{11.83}$$

This transform is the total sum in the rectangular brackets in (11.77). Substituting (11.83) for the terms in rectangular brackets in (11.77) simplifies that equation to

$$\sigma_d^2 A(\omega)\frac{1}{L}\sum_{m=0}^{L-1} G(\omega - m\omega_s)C(\omega - m\omega_s) + A(\omega) = \sigma_d^2 \tag{11.84}$$

so that

$$A(\omega) = \frac{\sigma_d^2}{\sigma_d^2 \frac{1}{L}\sum_{m=0}^{L-1} G(\omega - m\omega_s)C(\omega - m\omega_s) + 1} \tag{11.85}$$

Finally, the optimum equalizer transfer function is

$$\begin{aligned} H(\omega) &= G(\omega)A(\omega) \\ &= \frac{\overline{C(\omega)}}{S_{vv}(\omega)} \frac{\sigma_d^2}{\sigma_d^2 \frac{1}{L}\sum_{m=0}^{L-1} \frac{|C(\omega - m\omega_s)|^2}{S_{vv}(\omega - m\omega_s)} + 1} \end{aligned} \tag{11.86}$$

It is interesting to observe that the factor $\overline{C(\omega)}/S_{vv}(\omega)$ is a matched filter. It is also interesting to observe that the remaining factor has period ω_s with respect to ω. This follows from the fact that $C(\omega)$ and $S_{vv}(\omega)$ both have period $L\omega_s$. Therefore, this factor is equivalent to an infinite duration discrete-time filter with symbol spaced taps, that is, with spacing T.

11.2.2 MSE for the Optimum Linear Equalizer

The mean-squared error (MSE) is

$$\begin{aligned} \Lambda &= \mathrm{E}\left\{e(nT)\overline{e(nT)}\right\} = \mathrm{E}\left\{e(nT)\left[\overline{y(nT)} - \overline{d(nT)}\right]\right\} \\ &= \mathrm{E}\left\{e(nT)\overline{y(nT)}\right\} - \mathrm{E}\left\{e(nT)\overline{d(nT)}\right\} \end{aligned} \tag{11.87}$$

The down-sampled equalizer output $y(nT)$ is a linear function of the received signal samples $\{r\left(nT - i\frac{T}{L}\right)\}$ as can be seen in (11.52). Therefore, the first term on the right-hand side of (11.87) vanishes as a result of the orthogonality principle and the MSE for the optimum equalizer reduces to

$$\Lambda = -\mathrm{E}\left\{e(nT)\overline{d(nT)}\right\} = \sigma_d^2 - \mathrm{E}\left\{y(nT)\overline{d(nT)}\right\} \tag{11.88}$$

Substituting (11.53) for $y(nT)$ in this last equation, and using the facts that $d(nT)$ is an uncorrelated sequence and the noise $v(\cdot)$ is uncorrelated with $d(nT)$ gives

$$\mathrm{E}\left\{y(nT)\overline{d(nT)}\right\} = \sigma_d^2 \sum_{i=-\infty}^{\infty} h\left(i\tfrac{T}{L}\right) c\left(-i\tfrac{T}{L}\right) \tag{11.89}$$

According to Parseval's theorem for discrete-time Fourier transforms

$$\sum_{i=-\infty}^{\infty} h\left(i\tfrac{T}{L}\right) c\left(-i\tfrac{T}{L}\right) = \frac{1}{L\omega_s}\int_{\omega_0 - L\omega_s}^{\omega_0} H(\omega)C(\omega)\,d\omega \tag{11.90}$$

where ω_0 is a conveniently chosen constant. Since $H(\omega)$ and $C(\omega)$ have period $L\omega_s$ the integral can be taken over any interval of length $L\omega_s$. For example, we

can let $\omega_0 = \omega_s/2$ and then split the ω-axis up into L intervals of length ω_s to represent the integral as the following sum:

$$\frac{1}{L\omega_s}\int_{\omega_0-L\omega_s}^{\omega_0} H(\omega)C(\omega)\,d\omega = \frac{1}{L\omega_s}\sum_{m=0}^{L-1}\int_{\frac{\omega_s}{2}-(m+1)\omega_s}^{\frac{\omega_s}{2}-m\omega_s} H(\omega)C(\omega)\,d\omega$$

$$= \frac{1}{\omega_s}\int_{-\frac{\omega_s}{2}}^{\frac{\omega_s}{2}} \frac{1}{L}\sum_{m=0}^{L-1} H(\omega-m\omega_s)C(\omega-m\omega_s)\,d\omega \qquad (11.91)$$

Therefore, the MSE can be written as

$$\Lambda = \sigma_d^2\left[\frac{1}{\omega_s}\int_{-\frac{\omega_s}{2}}^{\frac{\omega_s}{2}} 1 - \frac{1}{L}\sum_{m=0}^{L-1} H(\omega-m\omega_s)C(\omega-m\omega_s)\,d\omega\right] \qquad (11.92)$$

Substituting the optimum equalizer transfer function (11.86) into (11.92), using the fact that the right-hand factor of $H(\omega)$ has period ω_s, and combining terms over a common denominator gives

$$\Lambda = \sigma_d^2\frac{1}{\omega_s}\int_{-\frac{\omega_s}{2}}^{\frac{\omega_s}{2}} \frac{1}{\sigma_d^2\frac{1}{L}\sum_{m=0}^{L-1}\frac{|C(\omega-m\omega_s)|^2}{S_{vv}(\omega-m\omega_s)}+1}\,d\omega \qquad (11.93)$$

Since the integrand has period ω_s, the limits can be changed to cover any convenient interval of length ω_s. For example, in a QAM system with carrier frequency ω_c it might be convenient to chose the interval to be $\omega_c - 0.5\omega_s < \omega < \omega_c + 0.5\omega_s$.

11.3 Finding the Initial Equalizer Taps by Using the FFT

In this section, we will see how to use a periodic training sequence like the V.34 PP sequence, which as period $N = 48$, and FFT methods to rapidly determine the initial equalizer coefficients. A sequence with period N can be represented as the sum of N sampled sinusoids at frequencies $k\omega_s/N$ for $k = 0,\ldots,N-1$ where $\omega_s = 2\pi/T$ is the sampling frequency and T is the sampling period. Therefore, the training sequence probes the channel at a discrete set of N frequencies. Assuming the channel noise is negligible, the receiver can determine the channel frequency response at the probe frequencies, since it knows the transmitted training sequence, by dividing the FFT of one period of the received sequence, suitably frequency translated to account for modulation

by a carrier and any channel frequency offset, by the FFT of one period of the training sequence. The signal-to-noise ratio in voiceband telephone channels is usually high enough to make this a good assumption. The receiver can then compute the optimum equalizer transfer function given by (11.51) at the probing frequencies and then take an inverse transform to find the equalizer taps. Details for this procedure are presented in the following subsections. Chevillat, Maiwald, and Ungerboeck [10] solved the problem of finding the N-tap FIR equalizer that perfectly equalizes the channel at the N probing frequencies in the noiseless case using a different method than in Section 11.2 and ended up with the same result as (11.51) sampled at the probing frequencies. They showed that this solution minimizes the squared norm of the equalizer impulse response, that is, the sum of the squared magnitudes of the N equalizer coefficients. They also presented methods for detecting the presence of a periodic training sequence and estimating the carrier frequency offset.

Suppose one period of the training sequence is $s_0,\ldots,s_{N-1}$. The discrete Fourier transform (DFT) of s_n is

$$S_k = \sum_{n=0}^{N-1} s_n e^{-j\frac{2\pi}{N}nk} \text{ for } k = 0,\ldots,N-1 \qquad (11.94)$$

If k is allowed to range beyond the set $\{0,\ldots,N-1\}$, the DFT repeats with period N. Therefore, for some negative index, $-k$, in the set $\{-N,\ldots,-1\}$,

$$S_{-k} = S_{N-k} \text{ with } 0 \le N-k \le N-1 \qquad (11.95)$$

The original sequence can be determined by the inverse discrete Fourier transform (IDFT) of S_k

$$\begin{aligned} s_n &= \frac{1}{N}\sum_{k=0}^{N-1} S_k e^{j\frac{2\pi}{N}nk} \\ &= \frac{1}{N}\sum_{k=0}^{N-1} S_k e^{j(k\frac{\omega_s}{N})nT} \text{ for } n = 0,\ldots,N-1 \end{aligned} \qquad (11.96)$$

The IDFT also has period N so this last equation actually holds for all n. This equation clearly shows that s_n is the sum of N sampled sinusoids at the frequencies $k\omega_s/N$ for $k = 0,\ldots,N-1$.

In V.34 modems the PP training sequence is a sequence of baseband QAM constellation points. These symbols are impulse modulated and applied to a baseband shaping filter with a square-root of raised cosine amplitude response with nominally an $\alpha = 0.12\%$ excess bandwidth [49, 60]. The shaping filter is a band-limited lowpass filter with a cut-off frequency of $(1+\alpha)\omega_s/2$. The impulse modulated PP sequence contains spectral components at the N frequencies

$k\omega_s/N$ for $k = 0, \ldots, N-1$ as well as the alias frequencies $k\omega_s/N - \ell\omega_s$ for all integer ℓ. Therefore, the frequencies that pass through the filter satisfy the inequality

$$-(1+\alpha)\frac{\omega_s}{2} \le k\frac{\omega_s}{N} \le (1+\alpha)\frac{\omega_s}{2} \tag{11.97}$$

or

$$-(1+\alpha)\frac{N}{2} \le k \le (1+\alpha)\frac{N}{2} \tag{11.98}$$

with the corresponding DFT values S_k and k adjusted for the DFT periodicity according to (11.95). The complex signal at the output of the shaping filter is then multiplied by the carrier $e^{-j\omega_c t}$ where ω_c is the carrier frequency, and the real part of the resulting signal is applied to the channel. The modulation translates the probing frequencies by the carrier frequency, so the channel is actually probed at the frequencies

$$\omega_c + k\frac{\omega_s}{N} \text{ for } -(1+\alpha)\frac{N}{2} \le k \le (1+\alpha)\frac{N}{2} \tag{11.99}$$

11.3.1 The Complex Cross-Coupled and Real Phase-Splitting Equalizers

Two types of equalizers are commonly used in telephone line modems: the complex cross-coupled equalizer shown in Figure 11.3, and the real phase-splitting equalizer shown in Figure 11.4 [60]. In the figures, the input signal $r(n)$ is a real passband QAM signal band limited to the interval

$$0 < \omega_c - (1+\alpha)\frac{\omega_s}{2} < |\omega| < \omega_c + (1+\alpha)\frac{\omega_s}{2} \tag{11.100}$$

where ω_c is the carrier frequency and, as usual, ω_s is the symbol rate. We will assume that the received signal is sampled at a rate $\omega_0 = L\omega_s$ that satisfies the Nyquist criterion for no aliasing, that is,

$$\omega_0 > 2\left[\omega_c + (1+\alpha)\frac{\omega_s}{2}\right] \tag{11.101}$$

and L is a convenient positive integer.

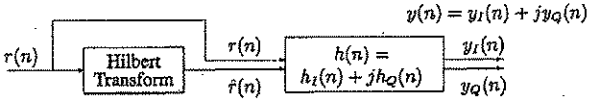


Figure 11.3. The Complex Cross-Coupled Equalizer

The Hilbert Transform box is a filter with the frequency response in the Nyquist band, $|\omega| < \omega_0/2$,

$$P(\omega) = -j\operatorname{sign}\omega = \begin{cases} -j & \text{for } 0 < \omega < \omega_0/2 \\ 0 & \text{for } \omega = 0 \\ j & \text{for } -\omega_0/2 < \omega < 0 \end{cases} \tag{11.102}$$

and which repeats with period ω_0 outside the Nyquist band. It is an ideal -90 degree phase shifter. The Hilbert transform of the received signal is

$$\hat{r}(n) = \sum_{k=-\infty}^{\infty} p(k)r(n-k) = p(n) * r(n) \tag{11.103}$$

It is convenient to consider the input to the complex cross-coupled equalizer to be the complex signal

$$r_+(n) = r(n) + j\hat{r}(n) \tag{11.104}$$

This signal is called the *pre-envelope* of $r(n)$. Over the Nyquist band, the discrete-time Fourier transform of the pre-envelope is

$$\begin{aligned} R_+(\omega) &= R(\omega) + j(-j\operatorname{sign}\omega)R(\omega) = 2R(\omega)u(\omega) \\ &= \begin{cases} 2R(\omega) & \text{for } 0 < \omega < \omega_0/2 \\ 0 & \text{for } -\omega_0/2 < \omega \le 0 \end{cases} \end{aligned} \tag{11.105}$$

The pre-envelope has no negative frequency components over the Nyquist band.

The output of the complex cross-coupled equalizer is the complex signal

$$\begin{aligned} y(n) &= h(n) * r_+(n) = [h_I(n) + jh_Q(n)] * [r(n) + j\hat{r}(n)] \\ &= [h_I(n) + jh_Q(n)] * [r(n) + jp(n) * r(n)] \end{aligned} \tag{11.106}$$

The real (inphase) and imaginary (quadrature) parts of $y(n)$ are

$$y_I(n) = r(n) * [h_I(n) - h_Q(n) * p(n)] \tag{11.107}$$

and

$$y_Q(n) = r(n) * [h_Q(n) + h_I(n) * p(n)] \tag{11.108}$$

These last two equations show that the inphase and quadrature components of the combined Hilbert transform and complex cross-coupled equalizer output can be generated by filtering the real input signal $r(n)$ by two sets of real coefficients. This observation suggests the real phase-splitting form of the equalizer shown in Figure 11.4. The coefficients for generating the inphase component are

$$\beta_n = h_I(n) - h_Q(n) * p(n) \tag{11.109}$$

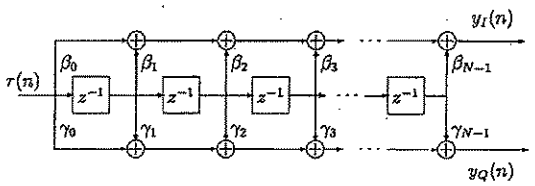


Figure 11.4. The Real Phase-Splitting Equalizer

and the coefficients for generating the quadrature component are

$$\gamma_n = h_Q(n) + h_I(n) * p(n) \tag{11.110}$$

In effect, the Hilbert transform has been incorporated into these two sets of coefficients. These sets of coefficients are also Hilbert transform pairs. The proof depends on the fact that $P^2(\omega) = -1$ so that $p(n) * p(n) = -\delta[n]$. Taking the Hilbert transform of β_n gives

$$\begin{aligned} \hat{\beta}_n &= p(n) * \beta_n = p(n) * h_I(n) - h_Q(n) * p(n) * p(n) \\ &= h_Q(n) + h_I(n) * p(n) = \gamma_n \end{aligned} \tag{11.111}$$

Let the output of the real phase-splitting equalizer be the complex signal $y(n) = y_I(n) + jy_Q(n)$. Then the transfer function over they Nyquist band from the input $r(n)$ to output $y(n)$ is

$$\begin{aligned} \tilde{H}(\omega) &= \sum_{n=-\infty}^{\infty} (\beta_n + j\gamma_n)e^{-j\omega nT_0} = \sum_{n=-\infty}^{\infty} (\beta_n + j\hat{\beta}_n)e^{-j\omega nT_0} \\ &= B(\omega) + j(-j\operatorname{sign}\omega)B(\omega) \\ &= 2B(\omega)u(\omega) \text{ for } |\omega| < \omega_0/2 \end{aligned} \tag{11.112}$$

Therefore, $\tilde{H}(\omega)$ is zero over the negative part of the Nyquist band. In terms of N-point FFT's, this means the FFT values are zero for $k = N/2, \ldots, N-1$.

In practice, the Hilbert transform filter can be approximated by an FIR filter, and a finite number of taps is selected for the equalizer that is adequate for equalizing the class of channels to be used. Therefore, a finite number of taps can be used for the inphase and quadrature sets as shown in Figure 11.4.

There are some practical considerations about sampling rates that have been glossed over so far. In practice, the Hilbert transform filter at the input to the complex cross-coupled equalizer would probably operate with T/4 sampling. Its output would be down-sampled by a factor of 2 to give a T/2 spaced equalizer, and the equalizer output would be down-sampled by a factor of 2 to get T spaced

output samples. The real phase-splitting equalizer might use T/3 spaced taps with the output down-sampled by a factor of 3 to get T spaced samples.

11.3.2 Computing the Equalizer Coefficients by Using the FFT

The steps required to rapidly compute the initial equalizer coefficients when a periodic training sequence is used are outlined in this subsection. The V.34 training signal contains six repetitions of the length $N = 48$ symbol CAZAC sequence presented in Section 11.1.3. This sequence probes the channel at a discrete set of frequencies, so the receiver can only determine the channel frequency response at these frequencies but perfect equalization can be achieved at them. The resulting equalizer will be a close approximation to the one required for random data transmission and can be used as a starting point followed by additional training using LMS adaptive training for an additional period of time. V.34 modems can use one of six different symbol rates ranging from 2400 to 3429 symbols/sec and two different carrier frequencies for each symbol rate except for the 3429 rate which can use only one carrier frequency. For each of these combinations, it is sufficient to sample the received signal at $L = 3$ times the symbol rate, ω_s, that is, with the sampling rate $\omega_0 = 3\omega_s$, to satisfy the Nyquist criterion. Therefore, many commercial modems use a real phase-splitting equalizer with $T/3$ tap spacing.

When using FFT methods to compute the equalizer coefficients, it is natural to use an FFT length equal to the number of samples in one period of the probe sequence. For L samples per symbol, the required length is $N_0 = LN$. For example, if $L = 3$ and $N = 48$, the required FFT length is $N_0 = 144$. We will see that the method for computing the equalizer taps requires the use of an N_0 tap equalizer.

Step 1: Frequency Translation to Align the Probe Frequencies on the DFT Frequencies

An N_0-point DFT computes spectral components at frequencies of the form $k\omega_0/N_0 = k\omega_s/N$ for $k = 0, \ldots, N_0 - 1$. Since the probe sequence is QAM modulated with a carrier frequency of ω_c and some frequency translation Δ can occur in the channel, the sinusoidal components of the received signal will not be at the DFT frequencies. Therefore, the first step at the receiver for both types of equalizers is to form the Hilbert transform $\hat{r}\left(n\frac{T}{L}\right)$ of the sequence $r\left(n\frac{T}{L}\right)$ and frequency translate the received signal to baseband by using the demodulation formula

$$x\left(n\tfrac{T}{L}\right) = \left[r\left(n\tfrac{T}{L}\right) + j\hat{r}\left(n\tfrac{T}{L}\right)\right] e^{-j(\omega_c+\Delta)nT/L} \tag{11.113}$$

Step 2: Data Collection

The next step at the receiver is to detect the start of the periodic training sequence and wait some time for transients to disappear. Then a record of $N_0 = NL$ samples of the received frequency translated signal, $x\left(n\frac{T}{L}\right)$ for $n = 0, \ldots, N_0 - 1$, can be collected. Since the periodic training sequence is repeated multiple times, several periods of collected points can be averaged to reduce noise effects.

Step 3: Form the N_0-Point FFT

Next compute the N_0-point DFT of the collected data record. The DFT formula is:

$$X_k = \sum_{n=0}^{N_0-1} x\left(n\tfrac{T}{L}\right) e^{-j\frac{2\pi}{N_0}nk} \text{ for } k = 0, \ldots, N_0 - 1 \qquad (11.114)$$

In practice, the DFT should be computed by a much faster FFT algorithm. For the V.34 case with $N_0 = 144 = 2^4 \times 3^2$, a special FFT can be designed using four radix 2 and two radix 3 stages.

Step 4: Computing the Optimum Equalizer Frequency Response

Let the N-point DFT of one period of the training sequence be $\{S_k;\ k = 0, \ldots, N\}$ as in (11.94) with the understanding that this sequence repeats with period N when k is outside the basic range. Then, the channel transfer function at the DFT frequencies can be computed as

$$C_k = \frac{X_k}{S_k} \text{ for } k = 0, \ldots, N_0 - 1 \qquad (11.115)$$

where $N_0 = NL$. Notice that the denominator, S_k, repeats L times as k ranges from 0 to $NL - 1$. Also, DFT indexes of the form $k + mN$ correspond to the frequencies $k\frac{\omega_s}{N} + m\omega_s$. According to (11.51) for the noiseless case, the optimum equalizer transfer function is

$$H_k = \frac{\overline{C}_k}{\frac{1}{L}\sum_{m=0}^{L-1}|C_{k+Nm}|^2} = \frac{\frac{\overline{X}_k}{\overline{S}_k}}{\frac{1}{L}\sum_{m=0}^{L-1}\left|\frac{X_{k+Nm}}{S_{k+Nm}}\right|^2}$$

$$= \frac{\frac{\overline{X}_k}{\overline{S}_k}}{\frac{1}{L|S_k|^2}\sum_{m=0}^{L-1}|X_{k+Nm}|^2} = \frac{\overline{X}_k S_k}{\frac{1}{L}\sum_{m=0}^{L-1}|X_{k+Nm}|^2}$$

$$= \frac{\overline{X}_k S_{\text{mod}(k,N)}}{\frac{1}{L}\sum_{m=0}^{L-1}\left|X_{\text{mod}(k,N)+Nm}\right|^2};\ k = 0, \ldots, NL - 1 \qquad (11.116)$$

The last form requires the least number of divisions. Notice that the denominator has period N with respect to k so it only has to be computed for $k = 0, \ldots, N-1$ and then repeated periodically. A small constant like σ_v^2/σ_d^2 in (11.49) can be added to the denominator to guarantee there is no overflow caused by division by a too small number.

For the V.34 case with $L = 3$ and $N = 48$, a slight simplification can be used because the transmitter's baseband shaping filter nominally has a low-pass square-root of raised cosine characteristic with $\alpha = 0.12$. The term $|X_{\text{mod}(k,N)+Nm}|^2$ for $m = 0$ covers the frequency range $[0, \omega_s)$ as k ranges from 0 to $N - 1 = 47$. The shaping filter squared amplitude response is down by a factor of 2 at $\omega_s/2$ and is zero beyond $(1 + \alpha)\omega_s/2$. The term for $m = 1$ covers the frequency range $[\omega_s, 2\omega_s)$ and the transmitted spectrum is identically zero in this range. The term for $m = 2$ covers the range $[2\omega_s, 3\omega_s)$ which aliases to $[-\omega_s, 0)$. Therefore, only the terms for $m = 0$ and 2 need to be included in the denominator sum.

Step 5: Finding the Equalizer Impulse Response

The next step is to find the initial N_0-point equalizer impulse response, h_n, by taking the inverse N_0-point FFT of H_k. The IDFT formula for the taps is

$$h_n = \frac{1}{N_0}\sum_{k=0}^{N_0-1} H_k e^{j\frac{2\pi}{N_0}nk} \text{ for } n = 0, \ldots, N_0 - 1 \qquad (11.117)$$

Step 6: Centering the Impulse Response

The resulting impulse response may not have its tap of peak magnitude near the center of the delay line corresponding to $n = N_0/2$ because of timing offsets between the transmitter and receiver assumed phase of the periodic training sequence. This still provides perfect equalization for the periodic probe signal but will not be good for random data. A result in DFT theory is that if an N_0-point time sequence is rotated cyclically to the right by some integer ℓ,

the N_0-point DFT of the rotated sequence is the original one with each term multiplied by the phase factor $\exp(-j2\pi\ell k/N_0)$. Therefore, the rotation will still leave the channel equalized but with the corresponding delay.

The location of the coefficient with the peak magnitude should be found and the coefficient sequence cyclically rotated to put the peak coefficient near the center point. This rotation should be by a complete number of symbols, that is, by a multiple of L positions, which we have assumed are spaced by T/L, to maintain the correct symbol timing. Suppose the coefficient sequence is rotated by $\ell = Li$ positions to the right. Then the DFT of the equalizer coefficient sequence becomes

$$H'_k = H_k e^{-j\frac{2\pi}{N_0}Lik} = H_k e^{-j\frac{2\pi}{N}ik} \tag{11.118}$$

The combined transfer function of the channel and equalizer before down-sampling is

$$\Gamma'_k = C_k H'_k = \frac{|C_k|^2}{\frac{1}{L}\sum_{m=0}^{L-1}|C_{k+Nm}|^2} e^{-j\frac{2\pi}{N}ik} \tag{11.119}$$

Using the fact that the denominator has period N with respect to k, the transfer function to the down-sampler output is

$$\Gamma_k = \frac{1}{L}\sum_{q=0}^{L-1}\Gamma'_{k+Nq} = \frac{\frac{1}{L}\sum_{q=0}^{L-1}|C_{k+Nq}|^2 e^{-j\frac{2\pi}{N}i(k+Nq)}}{\frac{1}{L}\sum_{m=0}^{L-1}|C_{k+Nm}|^2} \tag{11.120}$$

The complex exponential reduces to

$$e^{-j\frac{2\pi}{N}i(k+Nq)} = e^{-j\frac{2\pi}{N}ik}e^{-j2\pi iq} = e^{-j\frac{2\pi}{N}ik} \tag{11.121}$$

so the transfer function to the down-sampler output is just

$$\Gamma_k = e^{-j\frac{2\pi}{N}ik} \tag{11.122}$$

and the effect of the rotation is to introduce a delay of i complete symbols.

Step 7: Translation Back to Passband

The equalizer frequency response must be translated back to the actual carrier frequency. If the carrier frequency for the received signal is $\omega_c+\Delta$, the required passband equalizer coefficients are

$$h(n) = \begin{cases} h'_n e^{j(\omega_c+\Delta)n\frac{T}{L}} & \text{for } n = 0,\ldots,NL-1 \\ 0 & \text{elsewhere} \end{cases} \tag{11.123}$$

The transfer function of this complex passband equalizer is zero for $-L\omega_s/2 < \omega < 0$.

For the most rapid receiver training, the receiver's local carrier phase should be adjusted for the circular tap rotation of i symbols. However, if the carrier tracking loop is turned on after this fast equalization is completed, it will quickly rotate the signal constellation to the correct orientation.

Step 8: Computing the Real Phase-Splitting Equalizer Coefficients

The last step is to compute the coefficients for the real phase-splitting equalizer. The coefficient values given by (11.123) are for the received complex envelope $r_+\left(n\frac{T}{L}\right) = r\left(n\frac{T}{L}\right) + j\hat{r}\left(n\frac{T}{L}\right)$ which has the transform

$$R_+(\omega) = 2R(\omega)u(\omega) \text{ for } |\omega| < L\omega_s/2 \tag{11.124}$$

The transform of the equalized passband output is

$$Y(\omega) = R_+(\omega)H(\omega) = 2R(\omega)H(\omega)u(\omega) \tag{11.125}$$

If the real signal $r\left(n\frac{T}{L}\right)$ were the equalizer input, the output transform would be

$$Y(\omega) = R(\omega)H(\omega) = R(\omega)H(\omega)u(\omega) \tag{11.126}$$

because $H(\omega) = 0$ for negative frequencies in the Nyquist band. This is half the desired output. Therefore, the coefficients for the real phase-splitting equalizer should correspond to $2H(\omega)$ in the frequency domain and the required coefficients for the real phase-splitting equalizer are

$$\beta_n = 2\Re e\, h(n) \tag{11.127}$$

$$\gamma_n = 2\Im m\, h(n) \tag{11.128}$$

References

[1] AT&T, "Additional Details on AT&T's Candidate Modem for V.fast," CCITT Question 3/XVII, WP XVII/1, July 1991.

[2] AT&T, "ISI Coder – Combined Coding & Precoding," Telecommunications Industry Association (TIA), Subcommittee Contribution, TR30.1/93-06, Baltimore, MD, 14-18 June 1993.

[3] AT&T, Lei Fang Wei, "A New 4D 64-State Rate-4/5 Trellis Code," Telecommunications Industry Association (TIA), Subcommittee Contribution, TR30.1/93-07, Boston, 21-25 July 1993 (previously presented at the V.fast Rapporteur's Meeting, Rockville, MD, 12-14 May 1993.)

[4] Birkhoff, Garrett, and Saunders MacLane, *A Survey of Modern Algebra*, The Macmillan Company, New York, 1953,

[5] Blahut, Richard E., *Principles and Practice of Information Theory*, Addison Wesley, 1987.

[6] British Telecom PLC, "A Trellis Code for V.fast," CCITT, Study Group XVII, Question 3/XVII, V.fast Rapporteur's Group, Bath, England, 28-30, September 1992.

[7] British Telecom PLC, "A Trellis Code for V.fast," CCITT, Study Group XVII, Question 3/XVII, WP XVII/1, Delayed Contribution D240, Geneva, 12-15 January 1993.

[8] Calderbank, A.R., and Lawrence H. Ozarow, "Nonequiprobable Signaling on the Gaussian Channel," *IEEE Transaction on Information Theory*, Vol. 36, No. 4, July 1990, pp. 726–740.

[9] Calderbank, A.R., and N.J.A. Sloane, "New Trellis Codes Based on Lattices and Cosets," *IEEE Transactions on Information Theory*, Vol. IT-33, 1987, pp. 177–195.

[10] Chevillat, P.R., D. Maiwald, and G. Ungerboeck, "Rapid Training of a Voiceband Data-Modem Receiver Employing an Equalizer with Fractional-T Spaced Coefficients," *IEEE Transactions on Communications*, Vol. COM-35, No. 9, September 1987, pp. 869–876.

[11] Chevillat, P., and E. Eleftheriou, "Decoding of Trellis-Encoded Signals in the Presence of Intersymbol Interference and Noise," *IEEE Transactions on Communications*, Vol. COM-37, July 1989, pp. 669–676.

[12] Conway, J.H., and N.J A. Sloane, *Sphere Packings, Lattices, and Groups*, Springer-Verlag, New York, 1988.

[13] Coxeter, H.S.M., *Regular Polytopes*, Dover, N.Y., 3rd ed., 1973.

[14] Doob, J.L., *Stochastic Processes*, John Wiley & Sons, 1953.

[15] Duel-Hallen, A., and C. Heegard, "Delayed Decision-Feedback Equalization," *IEEE Transactions on Communications*, Vol. COM-37, May 1989, pp. 428–436.

[16] Eyuboğlu, M.V., and G.D. Forney, Jr., "Combined Coding and Equalization Using Trellis Precoding," presented at *Proc. CSI Workshop on Advanced Commun. Tech.*, Ruidoso, NM, May 1989.

[17] Eyuboğlu, M.V., and G.D. Forney, Jr., "Combined Coding and Equalization Using Trellis Precoding," presented at the *1990 Int. Information Theory Symposium*, San Diego, CA, January 14–19, 1990.

[18] Eyuboğlu, M.V., and G.D. Forney, Jr., "Trellis Precoding: Combined Coding, Precoding and Shaping for Intersymbol Interference Channels," *IEEE Transactions on Information Theory*, Vol. 38, No. 2, March 1992, pp. 301–313.

[19] Eyuboğlu, M.V., and S.U. Qureshi, "Reduced-State Sequence Estimation for Coded Modulation on Intersymbol Interference Channels," *IEEE Journal on Selected Areas in Communications*, Vol. SAC-7, August 1989, pp. 989–995.

[20] Forney, G. David, Jr., "Coset Codes—Part I: Introduction and Geometrical Classification," *IEEE Transactions on Information Theory*, Vol. 34, No. 5, September 1988, pp. 1123–1151.

[21] Forney, G. David, Jr., "Coset Codes—Part II: Binary Lattices and Related Codes," *IEEE Transactions of Information Theory*, Vol. 34, No. 5, September 1988, pp. 1152–1187.

[22] Forney, G. David, Jr., "Multidimensional Constellations II: Voronoi Constellations," *IEEE Journal on Selected Areas in Communications*, Vol. SAC-7, August 1989, pp. 941–958.

[23] Forney, G. David, Jr., "Trellis Shaping," *IEEE Transactions on Information Theory*, Vol. 38, No. 2, March 1992, pp. 281–300.

[24] Forney, G. David, Jr., and M.V. Eyuboğlu, "Combined Equalization and Coding Using Precoding," *IEEE Communications Magazine*, December 1991, pp. 25–34.

[25] Forney, G. David, Jr., and Lee Fang Wei, "Multidimensional Constellations—Part I: Introduction, Figures of Merit, and Generalized Cross Constellations," *IEEE Journal on Selected Areas in Communications*, Vol. 7, No. 6, August 1989, pp. 877–892.

[26] Forney, G. David, Jr., "Multidimensional Constellations – Part II," *IEEE Journal on Selected Areas in Communications*, Vol. 7, No. 6, August 1989, pp. 941–958.

[27] Gallager, R.G., *Information Theory and Reliable Communication*, John Wiley & Sons, New York, 1968.

[28] General DataComm, "Distribution-Preserving Tomlinson Algorithm," TIA/CCITT, Question 3/XVII, WP XVII/1, January 1992.

[29] General DataComm, Motorola Information Systems, Rockwell International, "Implementation of Precoding in V.fast," ITU-T working paper DT 17, Rockville, MD, May 12–14, 1993.

[30] General DataComm, Motorola Information Systems, Rockwell International, "Precoding for V.fast," Telecommunications Industry Association, TR-30/93-07-33 Committee Contribution, Newton, MA, 23 July 1993.

[31] Goldstein, Yuri, "Periodic Sequence for Fast Start Up Equalization," ITU-T, Question 1/14, WP 14/1, Rapporteur's Meeting, Dublin, IR, November 8–10, 1993.

[32] Harashima, Hiroshi, and Hiroshi Miyakawa, "Matched-Transmission Technique for Channels with Intersymbol Interference," *IEEE Transactions on Communications*, Vol. COM-20, No. 4, August 1972, pp. 774–780.

[33] ITU-T Recommendation V.32, "A Family of 2-Wire Duplex Modems Operating at Data Signalling Rates up to 9600 bit/s for use on the General Switched Telephone Network and on Leased Telephone-Type Circuits," www.itu.ch, March 1993.

[34] ITU-T Recommendation V.34, "A Modem Operating at Data Signalling Rates of Up to 33600 bit/s for use on the General Switched Telephone Network and on Leased Point-to-Point 2-Wire Telephone-Type Circuits," www.itu.ch, February 1998.

[35] ITU-T Recommendation V.90, "A Digital Modem and Analogue Modem Pair for use on the Public Switched Telephone Network (PSTN) at Data Signalling Rates of up to 56 000 bit/s Downstream and up to 33 600 bit/s Upstream," www.itu.ch, September 1998.

[36] ITU-T Recommendation V.92, "Enhancements to Recommendation V.90," www.itu.ch, November 2000.

[37] Johannesson, Rolf, and Kamil Sh. Zigangirov, *Fundamentals of Convolutional Coding*, IEEE Press, 1999.

[38] Kaplan, Wilfred, *Advanced Calculus*, Addison-Wesley, 1952.

[39] Khandani, A.K., and P. Kabal, "Shaping Multi-Dimensional Signal Spaces – Part I: Optimum Shaping, Shell Mapping," *IEEE Transactions on Information Theory*, Vol. 39, No. 6, November 1993, pp. 1799–1808.

[40] Khandani, A.K., and P. Kabal, "Shaping Multi-Dimensional Signal Spaces – Part II: Shell-Addressed Constellations," *IEEE Transactions on Information Theory*, Vol. 39, No. 6, November 1993, pp. 1809–1819.

[41] Kschischang, Frank Robert, "Shaping and Coding Gain Criteria in Signal Constellation Design," Ph.D. Thesis, University of Toronto, Canada, Department of Electrical Engineering, Communications Group Technical Report, June 1991.

[42] Lang, G.R., and F.M. Longstaff, "A Leech Lattice Modem," *Journal on Selected Areas in Communications*, Vol. 7, No. 6, August 1989, pp. 968–973.

[43] Laroia, Rajiv, "ISI Coder – Combined Coding & Precoding," TIA Subcommittee Contribution, TR30.1, Baltimore, MD, 14–18 June, 1993.

[44] Laroia, Rajiv, Nariman Farvardin, and Steven A. Tretter, "On SVQ Shaping of Multidimensional Constellations – High-Rate Large-Dimensional Constellations," *Proceedings*

of the 26th Annual Conference on Information Sciences and Systems, Princeton University, March 1992, pp. 527–531.

[45] Laroia, Rajiv, Nariman Farvardin, and Steven A. Tretter, "On Optimal Shaping of Multidimensional Constellations," *IEEE Transactions on Information Theory*, Vol. 40, No. 4, July 1994, pp. 1044–1056.

[46] Laroia, Rajiv, Nariman Farvardin, and Steven A. Tretter, "Precoding Scheme for Transmitting Data Using Optimally-Shaped Constellations Over Intersymbol-Interference Channels," U.S. Patent 5,388,124, February 7, 1995.

[47] Laroia, Rajiv, Steven A. Tretter, and Nariman Farvardin, "A Simple and Effective Precoding Scheme for Noise Whitening on Intersymbol Interference Channels," Proceedings of the 26th Annual Conference on Information Sciences and Systems, Princeton University, March 18–20, 1992, pp. 723–726.

[48] Laroia, Rajiv, Steven A. Tretter, and Nariman Farvardin, "A Simple and Effective Precoding Scheme for Noise Whitening on Intersymbol Interference Channels," *IEEE Transactions on Communications*, Vol. 41, No. 10, Oct. 1993, pp. 1460–1463.

[49] Lucky, R.W., J. Salz, and E.J. Weldon, Jr., *Principles of Data Communication*, McGraw-Hill, 1968.

[50] Milewski, A., "Periodic Sequences with Optimal Properties for Channel Estimation and Fast Start-Up Equalization," *IBM Journal Res. Develop.*, Vol. 27, No. 5, September 1983, pp. 426–429.

[51] Moon, Todd K. and Wynn C. Stirling, *Mathematical Methods and Algorithms for Signal Processing*, Prentice Hall, 2000.

[52] Motorola Information Systems, "Trellis Precoding for V.fast: Tutorial and Test Results," CCITT Question 3/XVII, WP XVII/1, April 1991.

[53] Motorola Information Systems, "A Flexible Form of Precoding for V.fast," CCITT working paper, Question 3/XVII, WP XVII/1, January 1992.

[54] Motorola Information Systems, "Signal Mapping and Shaping for V.fast," CCITT Delayed Contribution D 196, Question 3/XVII, WP XVII/1, Geneva, June 1992.

[55] Motorola Information Systems,"The 64-state 4D Wei code is quasicatastrophic," ITU-TSS, Question 3/14, WP 14/1, V.fast Rapporteur Group Meeting, Rockville, MD, USA, May 12-14, 1993.

[56] Pottie, G.J., and D.P. Taylor, "A Comparison of Reduced-Complexity Decoding Algorithms for Trellis Codes," *IEEE Journal on Selected Areas in Communications*, Vol. SAC-7, December 1989, pp. 1369–1380.

[57] Schlegel, Christian, *Trellis Coding*, IEEE Press, 1997.

[58] Tomlinson, M. "New Automatic Equaliser Employing Modulo Arithmetic," *Electronics Letters*, Vol. 7, Nos. 5/6, March 25, 1971, pp. 138–139.

[59] Tretter, Steven A., *Introduction to Discrete-Time Signal Processing*, John Wiley, 1976.

[60] Tretter, Steven A., *Communication System Design Using DSP Algorithms With Laboratory Experiments for the TMS320C30*, Plenum Press, 1995.

[61] Ungerboeck, G., and I. Csajka, "On improving data-link performance by increasing the channel alphabet and introducing sequence coding," 1976 Int. Symp. Information Theory, Ronneby, Sweden, June 1976.

[62] Ungerboeck, G., "Channel Coding with Multilevel/Phase Signals," *IEEE Transactions on Information Theory*, Vol. IT-28, No. 1, January 1982, pp. 55–67.

[63] Ungerboeck, G., "Trellis Coded Modulation with Redundant Signal Sets—Part I: Introduction and Part II: State of the Art," *IEEE Communications Magazine*, Vol. 25, No. 2, February 1987, pp. 5–21.

[64] Viterbi, Andrew J., "Convolutional Codes and Their Performance in Communication Systems," *IEEE Transactions on Communications Technology*, Vol. COM-19, No. 5, October 1971, pp. 751–771.

[65] Viterbi, Andrew J., and Jim K. Omura, *Principles of Digital Communication and Coding*, McGraw-Hill, 1979.

[66] Wei, Lee-Fang, "Trellis-Coded Modulation with Multidimensional Constellations," *IEEE Trans. on Information Theory*, Vol. IT-33, No. 4, July 1987, pp. 483–501.

[67] Wicker, Stephen B., *Error Control Systems for Digital Communication and Storage*, Prentice Hall, 1995.

[68] Wozencraft, J.M., and I.M. Jacobs, *Principles of Communication Engineering*, J. Wiley, 1965, pp. 355–357.

Index